



**UNIVERSIDAD
NACIONAL
DE LOJA**

TT-CIS



*Área de la Energía, las Industrias y los Recursos Naturales No
Renovables*

CARRERA DE INGENIERÍA EN SISTEMAS

“Prototipo de detección de personas autorizadas para encender un vehículo aplicando técnicas de visión artificial en dispositivos móviles Android.”

*“Trabajo de Titulación previo
a la Obtención del título de
Ingeniero en Sistemas”*

Director:

Ing. Mario Andrés Palma Jaramillo Mg. Sc.

Autor:

Jinsop Alexix Campos Paredes.

LOJA-ECUADOR

2015

CERTIFICACIÓN DEL DIRECTOR


Ing. Palma Jaramillo Mario Andrés, Mg. Sc.

DOCENTE DE LA CARRERA DE SISTEMAS, DEL AREA DE LA ENERGIA, LAS INDUSTRIAS Y DE LOS RECURSOS NATURALES NO RENOVABLES DE LA UNIVERSIDAD NACIONAL DE LOJA

CERTIFICO:

Que el presente trabajo de investigación elaborado por el señor **JINSOP ALEXIX CAMPOS PAREDES** titulado **“PROTOTIPO DE DETECCIÓN DE PERSONAS AUTORIZADAS PARA ENCENDER UN VEHÍCULO APLICANDO TÉCNICAS DE VISIÓN ARTIFICIAL EN DISPOSITIVOS MÓVILES ANDROID”**, ha sido dirigido, corregido y revisado prolijamente en su forma y contenido de acuerdo a las normas de graduación vigentes en el Reglamento del Régimen Académico de la Universidad Nacional de Loja, por lo que autorizo su presentación ante el respectivo Tribunal de Grado.

Loja, noviembre del 2015



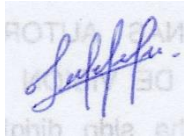
Ing. Palma Jaramillo Mario Andrés, Mg. Sc.
DIRECTOR DEL TRABAJO DE TITULACIÓN

AUTORÍA

Yo **JINSOP ALEXIX CAMPOS PAREDES** declaro ser autor 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.

Firma:



Cédula: 1105432163

Fecha: 18 de diciembre del 2015

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

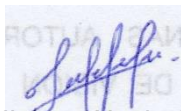
Yo **JINSOP ALEXIX CAMPOS PAREDES**, declaro ser autor de la tesis titulada: **PROTOTIPO DE DETECCIÓN DE PERSONAS AUTORIZADAS PARA ENCENDER UN VEHÍCULO APLICANDO TÉCNICAS DE VISIÓN ARTIFICIAL EN DISPOSITIVOS MÓVILES ANDROID**, como requisito para optar al grado de **INGENIERO EN SISTEMAS**; autorizo 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 éste 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, a los dieciocho días del mes de diciembre del dos mil quince.

Firma:



Autor: Jinsop Alexix Campos Paredes.

Cédula: 1105432163.

Dirección: Loja (Av. Eugenio Espejo y Holanda, sector Isidro Ayora).

Teléfono: 0989692874

Correo Electrónico: jinson.alexix@gmail.com

DATOS COMPLEMENTARIOS:

Director de Tesis: Ing. Mario Andrés Palma Jaramillo, Mg.Sc.

Tribunal de Grado: Ing. Luis Roberto Jácome Galarza, Mg.Sc.

Ing. Roberth Gustavo Figueroa Díaz, Mg.Sc.

Ing. Waldemar Victorino Espinoza Tituana, Mg.Sc.

DEDICATORIA

Dedico el presente trabajo de titulación a Dios sobre todas las cosas, por ser quien me permite vivir día a día, quien me ayuda a afrontar los momentos más difíciles, el que me guía para seguir adelante, a pesar de todo lo que se pueda presentar en mi diario vivir.

A mis padres José Campos y Herminia Paredes quienes me han dado la vida, me han brindado todo lo necesario y me han hecho todo lo que soy. A mis hermanas; Karina, Marisol, Patricia, mi hermano Edison, mi sobrina Allison y a toda mi familia, quienes aseguro se alegran por mis momentos de felicidad y me han brindado su apoyo incondicional en los buenos y malos momentos.

A mis amigos quienes considero mis hermanos los cuales han sido parte fundamental en mi vida; por su apoyo en los momentos de tristeza y alegría en los momentos de felicidad; amigos que perduraron durante todo este tiempo haciéndome saber que son verdaderos.

Jinsop Alexix Campos Paredes

AGRADECIMIENTO

Agradezco en primer lugar a Dios creador de todas las cosas, por darme la vida y cuidar de mí cada día. A mis padres José Campos y Herminia Paredes por haber sido unos padres responsables e incondicionales, por darme todo lo necesario para cumplir con mis metas, a mis hermanas Karina, Marisol, Patricia, mi hermano Edison, mi sobrina Allison y a toda mi familia por estar pendientes durante todo el desarrollo del presente trabajo de titulación; animándome en todo momento.

A la Universidad Nacional de Loja por brindarme la oportunidad de formarme como profesional; a los docentes que me han contribuido de forma positiva en mi educación; a mi director Ing. Mario Palma por guiarme y ser un apoyo fundamental durante el desarrollo del presente trabajo de titulación.

Y finalmente, de manera especial agradezco a mis amigos quienes han sido clave para la culminación del presente trabajo de titulación, debido a sus palabras de aliento y ánimo en todo momento, haciéndome saber que creen en mí.

Jinsop Alexix Campos Paredes

TABLA DE CONTENIDOS

CERTIFICACIÓN DEL DIRECTOR	ii
AUTORÍA	iii
CARTA DE AUTORIZACIÓN	iv
DEDICATORIA	v
AGRADECIMIENTO	vi
TABLA DE CONTENIDOS	vii
ÍNDICE DE FIGURAS	x
ÍNDICE DE TABLAS	xiii
a. TÍTULO	1
b. RESUMEN	2
ABSTRACT	3
c. INTRODUCCIÓN	4
d. REVISIÓN DE LITERATURA	6
CAPÍTULO I:	6
1. Computación móvil	6
1.1 Comunicaciones móviles	6
1.2 Hardware empleado en computación móvil	7
1.3 Sistemas Operativos móviles.....	11
CAPÍTULO II:	23
2. Visión Artificial	23
2.1 Introducción	23
2.2 Visión por computador en dispositivos móviles	28
2.3 OpenCV en Android.....	29
2.4 OpenCV en IOS.....	30
2.5 Técnicas de Reconocimiento facial.....	34
2.6 Eigenfaces.....	34
2.7 Fisherfaces	36
2.8 Patrones Binarios Locales Histogramas (LBPH)	38
CAPÍTULO III:	40

3. Herramientas de software y hardware seleccionadas	40
3.1 Herramientas de software	40
3.2 Herramientas de hardware.	46
CAPÍTULO IV:	53
e. MATERIALES Y MÉTODOS	53
1. Métodos.....	53
2. Técnicas	54
f. RESULTADOS	55
2.9 ANÁLISIS	55
1.1 Exploración.....	55
2. DISEÑO.....	61
2.1 Inicialización	61
3. CODIFICACIÓN.....	101
3.1 Producción y estabilización.....	101
4. PRUEBAS	110
4.1 Pruebas del sistema y arreglos.....	110
4.1.1.1 Caso de Prueba 1: Guardar dispositivo móvil.	110
g. DISCUSIÓN	131
1. Desarrollo de la propuesta alternativa.	131
1.1 Objetivo Específico 1: Determinar los requerimientos técnicos de Hardware y Software para la implementación del mecanismo de seguridad.	131
1.2 Objetivo Específico 2: Desarrollar la aplicación móvil android para detección de personas autorizadas.....	132
1.3 Objetivo Específico 3: Construir un prototipo de aplicación móvil android de reconocimiento facial conectada a la placa arduino para proteger el sistema de encendido del vehículo.	133
h. CONCLUSIONES	134
i. RECOMENDACIONES	135
j. BIBLIOGRAFÍA	136
k. ANEXOS	140
Anexo 1. Artículo científico	140
Anexo 2. Glosario de términos.....	149

Anexo 3. Licencia del Trabajo de Titulación.....	151
Anexo 4. Certificado de Resumen.....	152

ÍNDICE DE FIGURAS

Figura 1. Interior de una cámara web USB	9
Figura 2. Cámara en teléfono inteligente	10
Figura 3. Página de inicio de iOS.....	12
Figura 4. Pantalla de inicio de Android 4.4.....	14
Figura 5. Versiones de Android y su difusión	16
Figura 6. Arquitectura de Android.	17
Figura 7. Aspecto de la interfaz de Windows Phone	20
Figura 8. Espectrograma de la luz visible y no visible.	24
Figura 9. Ejemplo de cámara de visión artificial del sector industrial fabricada por Point-Grey.....	27
Figura 10. Adición del framework de OpenCV IOS en Xcode.	31
Figura 11. Cabeceras de trabajo en un fichero fuente IOS.....	32
Figura 12. Código para detección de caras.....	33
Figura 13. Resultado del ejemplo ejecutándose en el simulador.....	34
Figura 14. Interfaz de usuario de Eclipse Android.....	44
Figura 15. Interfaz de usuario de Fritzing.....	45
Figura 16. Interfaz de usuario de Arduino	46
Figura 17. Placa de Arduino uno.....	47
Figura 18. Arduino Mega 2560, vista delantera y posterior.	48
Figura 19. Módulo Bluetooth hc-06.	50
Figura 20. Módulo lector de SD.	51
Figura 21. Placa relé de control.	52
Figura 22. Directorio de instalación del NDK Android y OpenCV Android.	62
Figura 23. Selección de Directorio de trabajo.	62
Figura 24. Selección del Directorio del NDK Android.	63
Figura 25. Configuración de variable de entorno.....	63
Figura 26. Creación de un nuevo Proyecto Android.....	64
Figura 27. Conversión de proyecto a C/C++.	64
Figura 28. Final de Conversión de proyecto a C/C++.	65
Figura 29. Agregar librerías de OpenCV.	65

Figura 30. Escritura de comando de ejecución.	66
Figura 31. Selección de la variable de entorno NDKROOT.....	66
Figura 32. Diseño del Sistema.	71
Figura 33. Diagrama de clases	72
Figura 34. Diseño de base de datos.	73
Figura 35. Storyboard de la Aplicación	73
Figura 36. Prototipado de Pantalla: Asistente de configuración.	74
Figura 37. Prototipado de la Pantalla Detector Bluetooth.....	75
Figura 38. Prototipado de Pantalla: Instrucciones entrenamiento facial.	76
Figura 39. Prototipado de Pantalla: Asistente de configuración.	78
Figura 40. Prototipado de Pantalla: Datos personales.	79
Figura 41. Prototipado de Pantalla: Finalizar configuración.....	81
Figura 42. Prototipado de Pantalla: Dispositivos Arduino.....	82
Figura 43. Prototipado de Pantalla: Logueo facial.....	83
Figura 44. Prototipado de Pantalla: Logueo clave.....	85
Figura 45. Prototipado de Pantalla: Administrador.	86
Figura 46. Prototipado de Pantalla: Asistente de configuración.	88
Figura 47. Prototipado de Pantalla: Dispositivos móviles	89
Figura 48. Prototipado de Pantalla: Rostros de usuarios.	91
Figura 49. Prototipado de Pantalla: Asistente de configuración.	92
Figura 50. Prototipado de Pantalla: Asistente de configuración.	94
Figura 51. Esquema Conexión Arduino bluetooth.	96
Figura 52. Esquema conexión Arduino Lector SD.....	97
Figura 53. Esquema conexión Arduino Módulos Relay.	98
Figura 54. Esquema conexión Arduino Leds estado	99
Figura 55. Esquema final de conexiones.	100
Figura 56. Vista superior del prototipo.	100
Figura 57. Vista Lateral del prototipo.	101
Figura 58. Estructura de la aplicación móvil.	102
Figura 59. Código del splash	103
Figura 60. Código de envío y recepción de datos por Bluetooth.	104
Figura 61. Algoritmo MD5.	105
Figura 62. Código de detección de rostros.....	107

Figura 63. Código de entrenamiento de rostros.	109
Figura 64. Código de reconocimiento de rostros.....	110
Figura 65. Prueba unitaria: Guardar dispositivo móvil	111
Figura 66 .Ejecución de Prueba Unitaria: Guardar dispositivo móvil	111
Figura 67. Prueba unitaria: Cifrar datos	112
Figura 68 .Ejecución de Prueba Unitaria: Cifrar datos.....	112
Figura 69. Prueba unitaria: Enviar datos	113
Figura 70 .Ejecución de Prueba Unitaria: Enviar datos	114
Figura 71. Prueba unitaria: Logueo facial.....	115
Figura 72 .Ejecución de Prueba Unitaria: Logueo Facial.....	115
Figura 73. Prueba unitaria: Logueo clave.....	116
Figura 74 .Ejecución de Prueba Unitaria: Logueo Facial.....	116
Figura 75. Comprobación de datos ingresados: Datos personales campo vacío.	119
Figura 76. Comprobación de datos ingresados: Datos personales contraseña menor a 8 caracteres.....	120
Figura 77. Comprobación de datos ingresados: Datos personales contraseñas diferentes.	120
Figura 78. Prueba Funcional: Configuración Inicial.	125
Figura 79. Prueba funcional: Logueo Facial.....	127
Figura 80.Prueba funcional: Logueo clave.	129

ÍNDICE DE TABLAS

TABLA I. DISTRIBUCIÓN Y CRECIMIENTO DE SISTEMAS OPERATIVOS MÓVILES.	15
TABLA II. MARCOS DE TRABAJO NECESARIOS Y OPCIONALES PARA TRABAJAR CON OPENCV.	32
TABLA III. ESPECIFICACIONES TÉCNICAS ARDUINO MEGA 2560	49
TABLA IV. REQUERIMIENTOS FUNCIONALES.	56
TABLA V. REQUERIMIENTOS NO FUNCIONALES.	57
TABLA VI. MÓDULOS Y PROCESOS DE LA APLICACIÓN.	58
TABLA VII. HERRAMIENTAS SOFTWARE	59
TABLA VIII. HERRAMIENTAS HARDWARE.	60
TABLA IX. PLANIFICACIÓN DE FASES.	69
TABLA X. STORYCARD DE LA PANTALLA ASISTENTE DE INSTALACIÓN.	74
TABLA XI. STORYCARD DE LA PANTALLA DETECTOR BLUETOOTH.	75
TABLA XII. STORYCARD DE PANTALLA INSTRUCCIONES ENTRENAMIENTO FACIAL.	77
TABLA XIII. STORYCARD DE PANTALLA ENTRENAMIENTO FACIAL.	78
TABLA XIV. STORYCARD DE LA PANTALLA DATOS PERSONALES.	80
TABLA XV. STORYCARD DE LA PANTALLA FINALIZAR CONFIGURACIÓN.	81
TABLA XVI. STORYCARD DE PANTALLA DISPOSITIVO ARDUINO.	83
TABLA XVII. STORYCARD DE PANTALLA LOGUEO FACIAL.	84
TABLA XVIII. STORYCARD DE PANTALLA LOGUEO CLAVE	85
TABLA XIX. STORYCARD DE PANTALLA ADMINISTRADOR.	87
TABLA XX. STORYCARD DE PANTALLA ADMINISTRAR USUARIOS.	88
TABLA XXI. STORYCARD DE PANTALLA DISPOSITIVOS MÓVILES.	90
TABLA XXII. STORYCARD DE PANTALLA ROSTROS DE USUARIO.	91
TABLA XXIII. STORYCARD DE PANTALLA AJUSTES DE APLICACIÓN	93
TABLA XXIV. STORYCARD DE PANTALLA USUARIO.	94
TABLA XXV. MATERIALES HARDWARE	95
TABLA XXVI. CONEXIÓN DE PINES ARDUINO BLUETOOTH.	96
TABLA XXVII. CONEXIÓN DE PINES ARDUINO LECTOR SD	97

TABLA XXVIII. CONEXIÓN DE PINES ARDUINO MODULO RELAY 1	98
TABLA XXIX. CONEXIÓN DE PINES ARDUINO MODULO RELAY 2	98
TABLA XXX. CONEXIÓN DE PINES ARDUINO CON LED ACTIVIDAD.....	99
TABLA XXXI. CONEXIÓN DE PINES ARDUINO CON LED BLOQUEO	99
Tabla XXXII. CONEXIÓN DE PINES ARDUINO CON LED SEGUROS	99
TABLA XXXIII. PRUEBA UNITARIA: GUARDAR DISPOSITIVO MÓVIL.....	110
TABLA XXXIV. PRUEBA UNITARIA: CIFRAR DATOS.....	112
TABLA XXXV. PRUEBA UNITARIA: ENVIAR DATOS A TRAVÉS DE BLUETOOTH.	113
TABLA XXXVI. PRUEBA UNITARIA: GUARDAR DISPOSITIVO MÓVIL.	114
TABLA XXXVII. PRUEBA UNITARIA: LOGUEO USUARIO.....	115
TABLA XXXVIII. VERIFICACIÓN DE PANTALLAS.....	117
TABLA XXXIX. COMPROBACIÓN DATOS INGRESADOS: DATOS PERSONALES.	118
TABLA XL. VERIFICACIÓN DE REQUERIMIENTOS.....	121
TABLA XLI. PRUEBA FUNCIONAL: CONFIGURACIÓN INICIAL.	123
TABLA XLII. PRUEBA FUNCIONAL: LOGUEO CLAVE.....	126
TABLA XLIII. PRUEBA FUNCIONAL: LOGUEO CLAVE.....	128
TABLA XLIV. DISPOSITIVOS COMPATIBLES.	130

a. TÍTULO

“Prototipo de detección de personas autorizadas para encender un vehículo aplicando técnicas de visión artificial en dispositivos móviles android”.

b. RESUMEN

El presente trabajo de titulación trata acerca de la realización de una aplicación móvil Android de detección de usuarios autorizados para poder acceder a un vehículo determinado a través de reconocimiento facial, para lo cual utiliza y administra un dispositivo electrónico basado en la plataforma de hardware y software Arduino el que se instalará en el automovil.

Para su realización ha sido necesario abordar conceptos relacionados con aplicaciones móviles, sistema operativo android, conexión inalámbrica bluetooth, visión artificial, seguridad, electrónica y metodologías de desarrollo de aplicaciones móviles.

La metodología utilizada para el desarrollo de la aplicación ha sido Mobile-D, la cual es una metodología para conseguir ciclos de desarrollo muy rápidos en equipos muy pequeños, además está basada en metodologías conocidas pero aplicadas de forma estricta como: extreme programming, Crystal Methodologies y Rational Unified Process.

Entre los métodos que se usaron son el método científico, el método inductivo, el método deductivo, así también técnicas como es la observación directa y la revisión literaria todo esto englobado a un buen proceso de desarrollo.

Cada una de las fases realizadas dentro del presente proyecto ha permitido obtener los resultados necesarios para cumplir con cada uno de los objetivos planteados y así obtener una aplicación móvil que detectar y reconocer rostros como posible mejora de seguridad de las actuales alarmas de seguridad que se encuentran en los automóviles.

Para su desarrollo y construcción se ha integrado tecnologías de software como son Android, Java, C++, OpenCV Android, NDK Android; así como hardware, tablets Android, teléfonos Android y Arduino obteniendo buenos resultados en funcionamiento del prototipo construido.

ABSTRACT

The following degree work is about creating a mobile Android application in order to authorize and detect access to a particular vehicle through facial recognition, for which it uses and manages an electronic device based on a hardware and software platform Arduino (spanish word the name of a program), which will also be installed in the car.

For its realization it has been necessary to address the right concepts to mobile apps, android operating system, wireless connection, Bluetooth, artificial machine vision, security, electronic and methodologies for developing mobile applications.

The methodology used to develop this application is Mobile-D, this is a methodology for obtaining very fast cycles of development in very small devices, it is also based on known methodologies but they are strictly enforced as extreme programming, Crystal Methodologies and Rational Unified Process.

The following methods were used: scientific method, deductive method, and also techniques such as direct observation and literature review, everything encompassed to a good development process.

Each of the phases that were carried out within this project has allowed us to obtained the results needed in order to meet each of the objectives and obtain a mobile application to detect and recognize faces as a possible safety improvement from existing security alarms found in cars.

For its development and construction it has been integrated software technologies such as Android, Java, C++, OpenCV Android, Android NDK; as well as Android and Arduino phones, getting good results in performance from the prototype built.

c. INTRODUCCIÓN

Los dispositivos móviles se han convertido en los últimos años, en una de las tecnologías más utilizadas a diario por millones de personas alrededor de mundo, lo que ha hecho a estos dispositivos convertirse en una parte fundamental en el desarrollo cotidiano de cualquier actividad que realice una persona. Dicho esto, también se han ido acoplando con otros campos como son los autos, diseño gráfico, negocios, etc, con el fin de mejorar los servicios que estos ofrecen.

Considerando lo anterior se ha considerado tomar en cuenta los problemas de seguridad de la que disponen los vehículos o el uso que se le pueda dar en caso de que caigan en manos equivocadas, lo que hace necesario tener un mecanismo para protegerlo de accesos no autorizados a través de dispositivos móviles.

Por ello, se ha propuesto la realización del presente trabajo cuyo objetivo principal desarrollar un prototipo de aplicación móvil de visión artificial que brinde un mecanismo de seguridad a vehículos para restringir su uso por parte de personas no autorizadas mediante reconocimiento facial.

Las actividades que se realizaron en la resolución del mismo es determinar los requerimientos técnicos de Hardware y Software para la implementación del mecanismo de seguridad, desarrollar la aplicación móvil android para detección de personas autorizadas y por último construir un prototipo de aplicación móvil android de reconocimiento facial conectada a la placa arduino para proteger el sistema de encendido del vehículo.

Así mismo la estructura del proyecto se la realizó de acuerdo a los normativos establecidos como se detalla: el resumen presenta una síntesis del trabajo desarrollado que resalta los resultados obtenidos, los Índices correspondientes al contenido, figuras y tablas; la Introducción que contiene una descripción del trabajo realizado junto al contenido del presente documento.

Asimismo, también contiene la Revisión de Literatura dónde se ha incluido toda la información bibliográfica dividida en tres capítulos, los cuales son Computación móvil, Visión Artificial y Herramientas Software y Hardware.

Se ha incluido la sección denominada Materiales y Métodos utilizados, además contiene una breve información acerca de la metodología Mobile-d, la cual ha sido utilizada para realizar la aplicación Android.

En el apartado Resultados se incluyen todos los documentos, diagramas, tablas obtenidos a lo largo del desarrollo del presente trabajo de titulación, esto de acuerdo a las fases que se plantearon en el alcance del anteproyecto y de acuerdo a la metodología Mobile-d [1].

En la sección Discusión se hace un análisis acerca de los resultados obtenidos y su justificación como una propuesta alternativa a la problemática encontrada para la realización del presente trabajo.

El documento también contiene las Conclusiones y Recomendaciones obtenidas a través de la realización del proyecto; cuenta además con la bibliografía de la información recolectada en la revisión de literatura.

Por último, contiene los Anexos en dónde se encuentra el anteproyecto, y el artículo científico realizado en el presente trabajo de titulación.

d. REVISIÓN DE LITERATURA

La información bibliográfica recopilada para el desarrollo del presente proyecto de titulación se ha organizado en tres capítulos: Computación móvil, Visión artificial y Herramientas software y hardware.

CAPÍTULO I:

1. Computación móvil

La computación móvil es la tecnología que habilita el acceso a recursos digitales en cualquier momento y desde cualquier lugar. Representa un componente adicional conveniente a los sistemas distribuidos convencionales. En un sentido más amplio la computación móvil representa la eliminación de tiempo y espacio impuesta por los ordenadores de escritorio y las redes cableadas. [2]

1.1 Comunicaciones móviles

La computación móvil asume que es posible la interconexión sin hilos a redes de datos. Dicha comunicación emplea principalmente tres tipos de infraestructura [3]:

- **Servicios de telefonía móvil digital.** Emplean tecnologías como GSM/2G (Sistema Global para Comunicaciones Móviles), GRPS/2.5G (Servicio General de Paquetes Vía Radio), UMTS/3G (Sistema Universal de Comunicaciones Móviles) y LTE/4G la más reciente. Las distintas tecnologías difieren en su arquitectura, los protocolos y las bandas de frecuencia empleadas. Estas redes son accesibles gracias a varios operadores comerciales y están disponibles allí donde éstos últimos ofrecen cobertura. Su alcance se limita al área de cobertura asociada a una antena física del sistema (celda).
- **Redes locales inalámbricas de datos (WLAN).** Define a un conjunto de redes inalámbricas basadas en el estándar 802.11 definido por el IEEE (Institute of Electrical and Electronics Engineers). Estas redes también son conocidas como WiFi en virtud de

su promoción por la Wi-Fi Alliance que es una asociación comercial que certifica si un producto respeta el estándar y por tanto permite la interoperabilidad con otros productos certificados. Estas redes ofrecen tasas de transferencia dependientes del estándar empleado, pero en general son superiores a las ofrecidas mediante 3G, llegando en el caso del estándar 802.11g a los 54 Mbps. En este tipo de redes la tasa de transmisión también se ve afectada por las interferencias y la distancia que separa a los dispositivos interconectados.

- **Acceso a internet vía satélite.** Proporcionan acceso en aquellos lugares en los que no existe cobertura de redes móviles ni WiFi. Dicha infraestructura se basa en la retransmisión de señales de radio con una red de satélites en órbita geoestacionaria. Sus tasas de transferencia máxima llegan a 1 Gbps y su alcance depende de la visibilidad de los satélites de enlace.

1.2 Hardware empleado en computación móvil

Desde comienzo de la década de los noventa se ha ido presentando en el mercado diferentes tipos de dispositivos que responden a las características de movilidad mencionadas anteriormente y que no se corresponden con un computador personal portátil (*laptop PC*) [3]:

- **Asistentes digitales personales (PDA).** Se trata de un computador de bolsillo con una funcionalidad limitada. Está concebido como un complemento a un computador de escritorio con el que se sincroniza facilitando el acceso a la agenda de contactos, notas, correo electrónico y otras características.
- **Tabletas electrónicas (tablet).** Son computadores sin teclado y con un aspecto similar a una pizarra o bloc de notas con pantalla táctil y en los que a veces se incluye software de reconocimiento de escritura. Generalmente no están pensadas para aplicaciones en las que haya que hacer un uso intensivo del teclado (p. ej. procesamiento de texto) pero proporcionan mayor movilidad que la proporcionada por un computador portátil.

- **PC ultraportable (UMPC).** Es una especificación creada en el 2006 por Microsoft e Intel relativa al factor de forma de los computadores portátiles. Están concebidos como computadores de propósito general y se utilizan de modo similar a las tabletas electrónicas frente a las que han perdido en aceptación.
- **PC wearable.** Son dispositivos electrónicos en miniatura que pueden llevarse puestos bajo, con o sobre la ropa. Sus características son muy amplias y van desde pequeños dispositivos lúdicos personales a completos computadores integrados.
- **Computador a bordo (carputer).** Es una denominación que describe a los computadores móviles concebidos para funcionar e ir instalados en un automóvil con funciones de asistencia a la navegación (GPS) y gestión de los recursos de entretenimiento a bordo.
- **Teléfonos inteligentes (smartphones).** Son teléfonos de tamaño ligeramente superior al convencional que poseen un amplio rango de características y en los que puede instalarse aplicaciones para aumentar su funcionalidad. Pueden presentar teclado físico y/o pantallas sensibles al tacto.

1.2.1 Procesadores

La arquitectura ARM es una arquitectura de instrucciones RISC de 32 bits desarrollada por ARM Holdings (concebida originalmente por Acorn Computer Limited en 1983) que debido a su relativa simplicidad y bajo consume se ha convertido con el tiempo en la arquitectura predominante de los procesadores digitales presentes en los sistemas de computación móvil. Una consecuencia de las diferentes arquitecturas empleadas entre los sistemas de computación [3].

Una de las principales novedades soportadas en la arquitectura ARM es la presencia de múltiples núcleos como en las arquitecturas ARM Cortex-A9 MPCore y Snapdragon S4 [3].

1.2.2 Cámaras móviles

Debido a la importancia de las cámaras en el propósito de este proyecto, a continuación se realiza una revisión más detenida de los aspectos relacionados con las cámaras en los sistemas de computación móvil [3].

Las cámaras en los sistemas de computación móvil se pueden clasificar en dos grandes familias dependiendo de su propósito principal:

- **Cámaras web.** Nacen a principios de los noventa con el propósito de captar una imagen en movimiento para retransmitirla a través de redes digitales de datos. Algunas de sus principales aplicaciones son la videoconferencia y la videovigilancia. Están constituidas por una lente, un sensor de imagen, circuitería electrónica de soporte y, la mayoría, incluye también un micrófono. Aunque algunas son de enfoque fijo, presentan un sistema rudimentario de ajuste de enfoque mediante el giro manual del soporte de la lente (ver Fig. 1). Sin embargo, la limitación del enfoque en estas cámaras no afecta, hasta cierto punto, a la nitidez de la imagen debido a su gran profundidad de campo. En las cámaras de bajo coste el sensor de imagen suele emplear tecnología CMOS con una resolución VGA de 640x480 pixels a 30 fps (fotogramas por segundo). La electrónica de soporte capta la imagen del sensor de imagen transmitiendo cada fotograma al computador bien en formato RGB o YUV sin comprimir o JPEG comprimido.

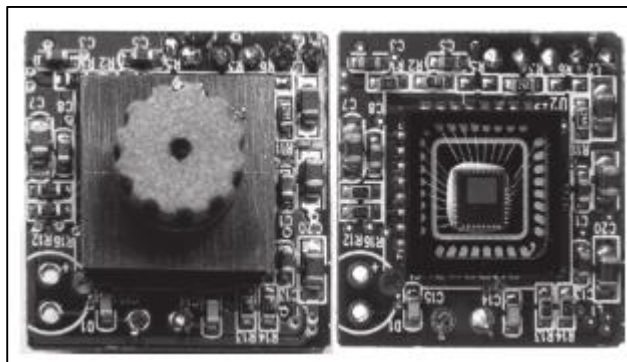


Figura 1. Interior de una cámara web USB

- **Cámaras fotográficas y de vídeo.** Este tipo de cámaras se concibe para su instalación en un teléfono móvil y aparecen por primera vez en el mercado hacia mediados de los

noventa. Inicialmente sus características diferían poco de aquellas presentes en las cámaras web (pequeño tamaño del sensor, enfoque fijo, pobre desempeño en condiciones de baja luminosidad). Sin embargo, con el paso del tiempo su calidad ha aumentado y sus especificaciones son cada vez más próximas a las de una cámara fotográfica digital compacta llegando a los 12 Mpx de resolución del sensor de imagen y HD1080p (1920x1080) para vídeo.

En la actualidad la presencia de una cámara (ver Fig. 2) es una de las señas de identidad de los teléfonos inteligentes y comienza a serlo igualmente en las tabletas electrónicas. Incluso la tendencia actual es la incorporación de dos cámaras, una trasera para la captura de foto fija y vídeo, y otra frontal más sencilla para funciones de videoconferencia.



Figura 2. Cámara en teléfono inteligente

Las cámaras móviles son cada vez más similares a las de cámaras digitales compactas dedicadas. En este sentido es común el prejuicio, potenciado interesadamente por las firmas comerciales, de que la calidad de una cámara es únicamente función de la resolución del sensor que incorpora. En realidad la calidad de una imagen es el resultado de una suma de factores como son: la resolución del sensor, el tamaño del sensor, la óptica empleada y la sensibilidad a la luz.

A continuación se realiza un repaso sobre aquellas características en las que aún existen diferencias importantes entre las cámaras móviles y las dedicadas [3]:

- **Tamaño del sensor de imagen.** Ya que en la cámaras móviles se intenta reducir el tamaño tanto como sea posible, los sensores son más pequeños que en las cámaras

dedicadas. Aunque existe un “guerra” comercial por ofertar cada vez más píxeles en el menor espacio posible, se debe tener presente que para el mismo tipo de sensor cuanto mayor es su tamaño mejores son el rango dinámico y la relación señal/ruido (SNR).

- **Óptica y enfoque.** La óptica de las cámaras móviles ha mejorado mucho desde que comenzó su comercialización, en la actualidad los fabricantes están incorporando ópticas de gran calidad, como Carl Zeiss. Siempre se trata de ópticas fijas pero se ha pasado del enfoque fijo al enfoque automático permitiendo incluso la fotografía de detalles a escasos centímetros (macro). Debido a las limitaciones de espacio, el zoom de las cámaras móviles no es óptico por lo que, en ningún caso, se gana nitidez en los detalles capturados mediante su uso.
- **Apertura.** En las cámaras móviles, la apertura es fija y la tendencia es emplear aperturas grandes persiguiendo mejorar la sensibilidad en condiciones de baja luminosidad.
- **Velocidad de obturación.** Las cámaras móviles carecen de obturador mecánico para reducir al máximo su tamaño por este motivo siempre es apreciable un indeseable retardo en la obturación. Esto unido a la menor sensibilidad de estas cámaras provoca que los tiempos de captura de la imagen sean superiores a los obtenidos con cámaras dedicadas haciendo casi imposible la captura de instantáneas de objetos en movimiento.
- **Disparo con flash.** Aunque ya es habitual la incorporación del flash, su desempeño es muy limitado aunque menos importante según mejora la sensibilidad en condiciones de baja luminosidad.

1.3 Sistemas Operativos móviles

Se denomina sistema operativo móvil al encargado de controlar un dispositivo móvil: PDA, tableta digital, teléfono inteligente, etc. Por tanto su aparición coincide con la comercialización de los primeros dispositivos móviles (teléfonos) aparecidos a finales de la década de los 70. Estos primeros SO son sistemas dedicados específicos para cada

dispositivo. Desde ese momento se han sucedido diferentes desarrollos con mayor o menor éxito.

Entre los principales sistemas operativos móviles tenemos:

1.3.1 IOS.

Este SO desarrollado por Apple Inc. en 2007 fue originalmente denominado iPhone OS, y diseñado para el control de dicho teléfono y el reproductor multimedia iPod touch. Posteriormente este SO ha extendido su soporte a la tableta iPad y la Apple TV. A diferencia de Android y Windows Phone, Apple no licencia el uso de iOS para hardware de otras compañías [4].

Este SO gestiona el hardware de los dispositivos y proporciona el conjunto de herramientas necesarias para desarrollar aplicaciones nativas empleando el lenguaje de programación Objective-C. Con el SO también se incluyen varias aplicaciones que proporcionan servicios básicos de usuario, como: Phone (gestión del teléfono), Mail (correo electrónico) y Safari (navegador web) (ver Fig. 3).



Figura 3. Página de inicio de iOS

Las aplicaciones desarrolladas con el SDK (*Software Development Kit*) de iOS se instalan en los dispositivos desde iTunes que es el gestor de contenidos de los dispositivos gobernados por iOS. Las aplicaciones se pueden descargar desde la tienda en línea de Apple (App Store) que en junio de 2012 disponía de un catálogo de más de 650.000 aplicaciones con un volumen de descargas totales de 30.000 millones hasta dicha fecha.

En conjunto el SDK de iOS consta de los siguientes componentes:

- **Xcode.** Entorno integrado de desarrollo (IDE) que proporciona las herramientas para el desarrollo de aplicaciones: edición, compilación, ejecución y depuración del proyecto en el que se construye una aplicación.
- **Instruments.** Herramienta de análisis de desempeño y depuración de la ejecución.
- **Simulador.** Es una aplicación de escritorio que permite ejecutar la aplicación iOS sin necesidad de emplear el dispositivo destinatario. Una particularidad del desarrollo de aplicaciones para la plataforma iOS es la necesidad de inscribirse en el programa de desarrollo de Apple si se desea probar las aplicaciones en un dispositivo físico.
- **Librería de desarrollo.** Son el conjunto de librerías que implementan las tecnologías empleadas en iOS y que durante el proceso de construcción son enlazadas estáticamente en la aplicación de usuario.

Es posible crear dos tipos de aplicaciones para iOS [5]:

- **Aplicaciones nativas.** Son aplicaciones de usuario que sólo son accesibles desde la pantalla del dispositivo. No es posible crear otro tipo de código como librerías dinámicas, frameworks, drivers, etc.
- **Aplicaciones web.** Son aplicaciones interactivas alojadas en un servidor web que se descargan a un cliente mediante la conexión de datos y se ejecutan dentro del

navegador Safari. No pueden ejecutarse sin el acceso al servidor donde se alojan. Pueden estar compuestas por código HTML, hojas de estilo (CSS) y JavaScript.

1.3.2 Android

La Open Handset Alliance (OHA) es un consorcio de más de 80 empresas, liderado por Google, constituido en 2007 y dedicado al impulso de estándares abiertos para dispositivos móviles. Desde su inicio la OHA ha abanderado el proyecto de desarrollo e implantación del SO móvil abierto Android. Google adquirió en 2005 la compañía de software Android Inc. embarcada en el proyecto de desarrollo del SO mencionado. En 2007, Google liberó bajo licencia Apache el código de Android, un SO (ver Fig. 4) para dispositivos móviles basado en Linux, para que todo aquel fabricante de hardware que lo deseara pudiera emplearlo en sus productos [6].



Figura 4. Pantalla de inicio de Android 4.4.

Para la comercialización de las aplicaciones Android, Google gestiona la tienda en línea *Google Play Market*. Mediante una aplicación preinstalada en Android el usuario puede descargar las aplicaciones compatibles publicadas en la tienda.

A finales de 2010 las cifras convirtieron a Android en la plataforma líder a nivel mundial en el mercado de los teléfonos inteligentes. Su cuota de mercado en el primer cuatrimestre de 2012 alcanzó el 56 % con 331 millones de dispositivos en funcionamiento. En la Tabla I se muestra un resumen de los datos recogidos en un estudio [7].

TABLA I. DISTRIBUCIÓN Y CRECIMIENTO DE SISTEMAS OPERATIVOS MÓVILES.

	España		Mundo *	
	ago-12	Crec.	ago-12	Crec.
Android	86,8%	29,1%	61,2%	9,0%
RIM	6,3%	-6,7%	3,8%	-4,3%
Apple	2,9%	-5,1%	23,7%	0,4%
Symbian	2,3%	-16,6%	3,2%	-6,7%
Microsoft	1,4%	-1,0%	4,8%	1,2%
Otros	0,3%	0,3%	3,4%	0,4%

Una de las principales características de Android es que se trata de un proyecto de código abierto que puede utilizar cualquier fabricante de hardware e incluso personalizar a sus necesidades. Esta cualidad de Android permite que exista compatibilidad de las aplicaciones con los dispositivos certificados en el programa de compatibilidad de Android.

La principal plataforma hardware para Android es la arquitectura ARM aunque existe un proyecto para ofrecer soporte a la arquitectura x86 (Android x86). Desde el lanzamiento de la primera versión denominada «Astro» se han ido sucediendo regularmente nuevas versiones (ver Fig. 5) que añaden nuevas características y corrigen errores.

Version	Codename	API	Distribution
2.2	Froyo	8	0.2%
2.3.3 - 2.3.7	Gingerbread	10	4.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	3.7%
4.1.x	Jelly Bean	16	12.1%
4.2.x		17	15.2%
4.3		18	4.5%
4.4	KitKat	19	39.2%
5.0	Lollipop	21	15.9%
5.1		22	5.1%

Figura 5. Versiones de Android y su difusión

1.3.2.1 Arquitectura de Android

Android consta de un núcleo (kernel) basado en núcleo de Linux empleando un middleware, librerías y API programadas en C. Las aplicaciones se ejecutan en un contexto Java que incluye librerías compatibles con Apache Harmony que es una implementación de Java de código abierto [8].

Aunque el núcleo de Android está basado en Linux (ver Fig. 6) no existe compatibilidad entre las aplicaciones para ambos SO ya que Android no soporta ni el sistema X Windows, ni el conjunto completo de librerías GNU.



Figura 6. Arquitectura de Android.

Mediante del framework de aplicaciones los desarrolladores pueden construir nuevas aplicaciones en lenguaje Java empleando las mismas librerías que las aplicaciones nativas.

El empleo de una arquitectura para las aplicaciones simplifica la reutilización de componentes de otras aplicaciones. Por ejemplo, una aplicación puede emplear el gestor de contactos para seleccionar un contacto determinado.

Toda aplicación Android se construye en base a los componentes fundamentales siguientes [8]:

- **Actividad (Activity).** Constituye la capa de presentación de la aplicación. De modo sencillo puede decirse que es cada una de las pantallas que se presentan al usuario.
- **Vista (View).** Son cada uno de los widgets de la interfaz de usuario, como botones, campos de texto, etc. La clase base de todas las vistas es `android.view.View` cuyos atributos pueden modificarse para alterar el aspecto visual y comportamiento del widget. Las vistas pueden agruparse mediante el gestor de layout con la clase base `android.view.ViewGroup`.
- **Intenciones (Intents).** Son mensajes asíncronos que permiten a una aplicación solicitar funcionalidad de otro componente del sistema (p. ej. de servicios o actividades). Para hacer uso de ellos se puede llamar explícitamente a un componente o bien solicitar al sistema la evaluación de un componente registrado para responder a la solicitud (solicitud implícita).
- **Servicios.** Realizan tareas en segundo plano sin proporcionar interfaz de usuario. Pueden enviar notificación al usuario mediante el framework asociado a las notificaciones.
- **Proveedores de contenido (ContentProvider).** Proporciona una interfaz estructurada para los datos de la aplicación. Mediante este componente se facilita la compartición de datos entre aplicaciones. Junto con este mecanismo Android proporciona una base de datos SQLite para la implementar la persistencia de datos.
- **Receptor de retransmisión (BroadcastReceiver).** Este componente puede registrarse para recibir mensajes tanto del sistema como intenciones de otra aplicación. Este componente recibe una notificación siempre que se produzca la situación asociada a dicho receptor. Por ejemplo, el receptor puede recibir una notificación cuando el terminal recibe una llamada telefónica.
- **Widgets.** Son componentes interactivos que se emplean principalmente en la pantalla de inicio (home screen) de Android. En ellos se muestra información al usuario (p. ej. número de mensajes sin leer) y permiten interactuar con la aplicación.

Todas las aplicaciones para Android se programan usando el lenguaje Java y se compilan empleando las herramientas suministradas con el SDK de Android para generar un archivo con la extensión .apk que contiene el código de la aplicación. Una vez que la aplicación (archivo .apk) se instala en el dispositivo, su «ciclo de vida» se circunscribe a un contexto de seguridad individual impenetrable (sandbox). En este sentido Android proporciona los mecanismos que se señala a continuación [8]:

- El SO (multiusuario) asocia un usuario a cada aplicación con un identificador único (ID) que ni siquiera es conocido por la aplicación sino únicamente por el sistema.
- El sistema asigna el mismo ID a los ficheros que son accesibles por la aplicación. De este modo se asegura que el acceso a los datos sólo es posible desde la propia aplicación. El código de cada aplicación se ejecuta de modo aislado en su propia máquina virtual.
- Cada aplicación se ejecuta, por defecto, en un proceso Linux independiente que se inicia cada vez que uno de los componentes de la aplicación necesita ejecutarse.

Los mecanismos anteriores llevan a cabo el denominado «principio de privilegio mínimo» por el cual, si no se indica lo contrario, una aplicación sólo tiene acceso a los componentes precisos para desarrollar su función.

Sin embargo, también se proporcionan mecanismos para que las aplicaciones puedan compartir información entre sí y para que éstas puedan acceder a los servicios del sistema:

- Dos aplicaciones pueden compartir el mismo ID de usuario de modo que puedan compartir los ficheros accesibles por cada una de ellas. Estas aplicaciones también se ejecutan como un único proceso Linux compartiendo una única instancia de la máquina virtual, por lo que es necesario las aplicaciones estén firmadas con un único certificado.

- Una aplicación puede solicitar permiso de acceso a los datos del dispositivo como la lista de contactos del usuario, los mensajes SMS, el almacenamiento en tarjeta SD, la cámara, bluetooth, etc. El permiso de acceso a estos datos se otorga en el proceso de instalación de la aplicación.

1.3.3 Windows Phone

Windows Phone (ver Fig. 7) (actualmente en la versión 10, Windows 10 for phones) es un SO desarrollado por Microsoft que sucede a la plataformas previa Windows Mobile, aunque es incompatible con ella. Su objetivo, en lugar del empresarial, es el mercado de consumo donde el volumen de negocio ha crecido de modo espectacular en los últimos años. Para facilitar la distribución de contenido multimedia y aplicaciones de terceros también se ha abierto una tienda en línea denominada Windows Phone Marketplace y gestionada por Microsoft [9].



Figura 7. Aspecto de la interfaz de Windows Phone

El desarrollo de las aplicaciones para Windows Phone se basa en herramientas y tecnologías de Microsoft existentes (Visual Studio, Expression Blend, Silverlight y el framework XNA) lo que facilita mucho la tarea de los programadores familiarizados con las mismas.

La plataforma de aplicaciones Windows Phone proporciona dos frameworks para el desarrollo de aplicaciones:

- **Silverlight.** Es un subconjunto de WPF (*Windows Presentation Foundation*) para el desarrollo de aplicaciones dirigidas por eventos basadas en XAML (*eXtensible Application Markup Language*).
- **XNA.** Para el desarrollo de juegos.

1.3.3.1 Arquitectura de la plataforma de aplicaciones Windows Phone

La arquitectura consta de cuatro componentes [10]:

- **Ejecutables (Runtimes).** Con los frameworks Silverlight y XNA todo el desarrollo se realiza en un entorno protegido en que se requieren muy pocos ajustes para la ejecución en un dispositivo concreto. Los dos frameworks mencionados junto con los componentes específicos de Windows Phone y la librería base de clases comunes (Common Base Class Library) proporcionan el conjunto de componentes para el desarrollo de las aplicaciones. Silverlight proporciona los componentes para la creación de aplicaciones con una interfaz de usuario basada en páginas. XNA consta de software, servicios y recursos accesibles mediante las correspondientes API cuyo objetivo es el desarrollo de juegos en las plataformas destinadas a tal fin.
- **Herramientas.** Para desarrollar aplicaciones es preciso instalar el SDK para Windows Phone. Está disponible de modo gratuito desde una dirección web de Microsoft.
 - Visual Studio 2010 es el IDE para construir aplicaciones Windows Phone (junto con el SDK se suministra la versión Visual Studio 2010 Express para Windows Phone).
 - Expression Blend es una herramienta para el diseño de aplicaciones web que incluso pueden emplear la tecnología Silverlight.
 - Emulador de Windows Phone que puede emplearse desde Visual Studio y Expression Blend para testear una aplicación y depurarla de un modo más

sencillo y eficiente. Incluye emulación de GPU y cambios de orientación de la pantalla.

- XNA Game Studio. Es una plataforma integrada de desarrollo de juegos para la gama de dispositivos de Microsoft dirigidos al ocio. Ejemplos, documentación, guías y comunidad.

CAPÍTULO II:

2. Visión Artificial

2.1 Introducción

La visión artificial o visión por computador es la ciencia y la tecnología que permite a las "máquinas" ver, extraer información de las imágenes digitales, resolver alguna tarea o entender la escena que están visionando [11].

Actualmente, las aplicaciones de la visión artificial están muy extendidas y van desde el campo de la industria (contar botellas, comprobar defectos en una cadena de montaje, interpretar un TAC médico...) y el campo de la medicina (recuento y búsqueda de células), hasta los sistemas más complejos, que permiten a los robots orientarse en un entorno desconocido, pasando por el reconocimiento de patrones de la realidad aumentada, entre otras muchas aplicaciones.

2.1.1 Luz visible, luz invisible

El ojo humano ve una parte del espectro de toda la luz que ilumina el universo. El rango de luz que podemos ver se lo denomina luz visible. Esto quiere decir que hay frecuencias de luz que no podemos ver pero que existen, como por ejemplo, los infrarrojos y los ultravioletas [11].

Los infrarrojos son los "colores" no visibles al ojo humano que están por debajo del rojo desde el punto de vista frecuencial.

Los ultravioletas son los "colores" no visibles al ojo humano que están por encima del violeta (ver Fig. 8).

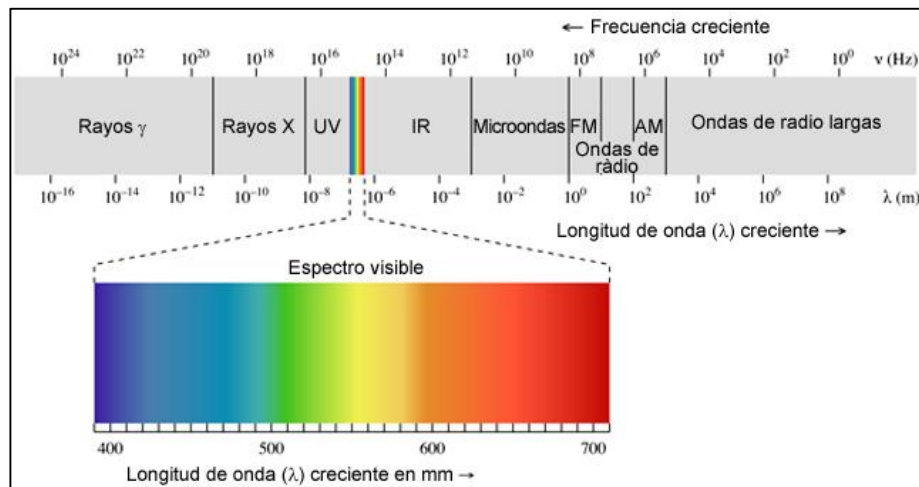


Figura 8. Espectrograma de la luz visible y no visible.

El hecho es que la mayoría de sensores de cámaras digitales son sensibles a la luz visible, pero también son sensibles (en diferente medida) a la luz infrarroja y/o a la ultravioleta.

Ahora bien, la luz infrarroja no nos es útil para construir una imagen digital, puesto que nos da información de una frecuencia que no podemos ver y que, por lo tanto, no tiene una representación posible en un color.

Por esta razón, la mayoría de cámaras digitales enfocadas a hacer fotografías o fotogramas llevan un filtro anti-IR, o sea antiinfrarojos, para cortar todas las frecuencias por debajo del espectro visible que nos resultarían un ruido innecesario, y solo dejan pasar el rango de luz visible que queremos plasmar con la cámara.

2.1.2 La imagen en movimiento

Los procesos de visión artificial son posibles gracias a tecnologías basadas en la captura de la imagen (cámaras de vídeo, cámaras web), sumadas a la capacidad de procesamiento de los ordenadores actuales [11].

Toda una serie de fotografías disparadas a una gran frecuencia que, reproducidas después a esa misma velocidad, provocan en los espectadores la ilusión del movimiento.

Actualmente, la mayoría de tecnologías de captación de imagen en movimiento (videocámaras) funcionan mediante el uso de sensores electrónicos (CCD y CMOS).

Durante la primera década del siglo XXI, se ha estandarizado el uso de la fotografía y el vídeo digitales, lo que permite la grabación de imágenes en alta resolución a un relativo bajo coste y con un elevado número de imágenes por segundo (FPS).

2.1.3 La imagen digital

2.1.3.1 Matrices de píxeles

En el mundo digital, las imágenes se representan como una matriz bidimensional de píxeles en la que cada píxel puede adquirir valores de color codificados con tres parámetros (R: red, G: green, B: blue) [12].

A pesar de que ello se ha heredado del mundo de la imagen analógica, normalmente trabajaremos con imágenes de proporción 4 × 3 (cuatro unidades de anchura por tres unidades de altura); es el caso de resoluciones estándar como 640 × 480 px, 800 × 600 px y 1.024 × 768 px.

Frecuentemente, se utilizara imágenes de baja resolución (entre 160 × 120 y 640 × 480 píxeles) como fuente de análisis de los procesos de visión artificial, ya que en la mayoría de casos no se necesita excesivo detalle en las imágenes para efectuar un análisis orientado a la visión por computador.

2.1.3.2 Bytes, bits y colores

Un píxel normalmente se expresa mediante tres números enteros (R, G, B), que representan los componentes rojo, verde y azul de todo color. Estos valores de R, G y B se suelen expresar en un rango de 8 bits, o sea, de valores entre 0 y 255 [12].

Por ejemplo, un píxel que tenga unos valores RGB de (255, 0, 0) nos indica un color rojo puro. Un píxel que tenga unos valores RGB (255, 0, 255) nos indica un color producido por una mezcla del rojo puro y el azul puro; en este caso, por lo tanto, obtendremos un color lila intenso.

2.1.3.3 Frecuencia de imagen (frame rate)

La frecuencia de imagen (frame rate) hace referencia al número de imágenes por segundo. Es la medida de la frecuencia a la que un reproductor de imágenes muestra diferentes fotogramas (frames) [13].

En informática estos fotogramas están constituidos por un número determinado de píxeles que se distribuyen a lo largo de una red de texturas. La frecuencia de los fotogramas es proporcional al número de píxeles que se han de generar y que inciden en el rendimiento del ordenador que los reproduce.

La frecuencia de actualización de las imágenes oscila, en el entorno digital, entre los 15 y los 60 FPS (frames por segundo). El rango entre 25 y 30 FPS es el más común. Según nuestras necesidades, deberemos elegir tecnologías de captación de la imagen con una frecuencia de imagen específica.

Si queremos analizar y extraer datos de objetos o usuarios que se mueven a grandes velocidades (coches, pájaros, corredores de fútbol...), seguramente necesitaremos un sistema de captación de vídeo con más resolución temporal (una frecuencia de imagen más alta: más fotogramas por segundo).

2.1.4 Herramientas para la Visión Artificial

2.1.4.1 La cámara

Para empezar a trabajar con la visión artificial, necesitamos un dispositivo sensible a la luz visible que nos permita almacenar las imágenes en formato digital. En otras palabras, necesitamos una cámara web, una cámara de vídeo o una capturadora analógica (ver Fig. 9).



Figura 9. Ejemplo de cámara de visión artificial del sector industrial fabricada por Point-Grey

Las cámaras web y la mayoría de cámaras de vídeo se pueden conectar directamente a los ordenadores por medio de los puertos USB, FireWire o Thunderbolt, y podemos capturar sus fotogramas en tiempo real [14].

En todo caso, también podemos utilizar una capturadora analógica, que a partir de una señal de vídeo analógico, nos permitirá capturar imágenes y procesarlas.

Una vez que ya tengamos el dispositivo de captura en funcionamiento, hemos de considerar toda una serie de aspectos referentes al entorno en el que se hará la interacción, puesto que la variación en la iluminación, la complejidad de la imagen u otros factores pueden dificultar mucho el proceso de visión artificial.

El proceso de "enseñar" a un ordenador a tomar decisiones por medio de la "visión" implica muchas dificultades y, por lo tanto, es muy importante trabajar en entornos en los que la luz y el escenario que se quieran analizar sean cuanto más sencillos y estables mejor [14].

2.1.4.2 Iluminación de la escena

La adquisición de imágenes por parte de una cámara varía mucho según la iluminación de la escena. Un cambio lo bastante fuerte en el ambiente lumínico puede hacer que todo el sistema de visión artificial funcione de un modo muy diferente.

En casos en los que no se pueda conseguir un ambiente estable (como por ejemplo una aplicación al aire libre), deberemos ir adaptando de alguna manera el sistema de adquisición de imágenes a las condiciones cambiantes mediante algoritmos adaptativos que permitan analizar periódicamente los cambios de iluminación del entorno y adaptar el sistema de visión a las nuevas condiciones [15].

Algunas cámaras, como la que lleva el mando de la Wii o muchas del sector industrial, son especialmente sensibles a los infrarrojos, puesto que nos permiten trabajar en un entorno sin luz visible, en la oscuridad, siempre que tengamos alguna fuente de luz infrarroja.

2.2 Visión por computador en dispositivos móviles

Tradicionalmente, la visión por computador es una disciplina científica que ha requerido equipos con gran poder de cómputo. Gracias a los espectaculares avances de la tecnología, los teléfonos móviles de última generación han adquirido capacidades que permiten la ejecución de programas de procesamiento de imagen [16].

Muchas empresas y programadores han empezado a explotar comercialmente aplicaciones de visión en dispositivos móviles. Debe tenerse en cuenta que estas aplicaciones también se aprovechan del mercado y audiencia potencialmente mundiales que hoy en día constituye la base de distribución de aplicaciones en dispositivos móviles.

El interés es también muy grande a nivel académico y científico [17]. Los dispositivos móviles permiten crear aplicaciones de visión por computador muy diversas, algunos ejemplos son:

- Interfaces gestuales
- Reconocimiento óptico de caracteres

- Búsqueda visual
- Realidad aumentada
- Aplicaciones avanzadas de fotografía
- Vigilancia

La librería de funciones más conocida y potente para visión por computador es OpenCV [18]. En la actualidad OpenCV está soportada por las empresas Willow Garage e Itseez y dispone de más de 2.500 algoritmos. En el momento de escribir estas líneas existe una versión de OpenCV para Android, para Windows e IOS.

2.3 OpenCV en Android

Como ya se comentó en la Sección anterior 2.2 OpenCV es la librería más conocida y potente para el desarrollo de aplicaciones de Visión Artificial y por tanto de máximo interés para el desarrollo de Apps que emplean dicha tecnología.

Desde la versión 2.3.0 de dicha librería está disponible un SDK OpenCV para Android (OpenCV4Android SDK) con el que se suministra el código binario de la librería para las plataformas ARM-v5, ARM-v7a, MIPS y x86. Existen dos modos de uso de OpenCV en las aplicaciones Android [19]:

1. **Java.** En este modo se emplean las clases Java suministradas por el SDK OpenCV4Android para acceder a la funcionalidad de la librería, que es tratada como un proyecto de librería, incluyéndola en el proyecto Android se desarrolle.
2. **Nativo C/C++.** En esta variante se accede a dichas librerías empleando la funcionalidad suministrada por NDK.

A partir de la versión OpenCV4Android 2.4.3 se introdujo la inicialización de la librería mediante la API OpenCV Manager que introduce notables mejoras en el desarrollo de las aplicaciones [20]:

- Las aplicaciones resultantes son más compactas pues no es preciso que contengan el código correspondiente a la librería.
- Se hace uso automático de las optimizaciones que existan para una plataforma específica de destino.

- Las actualizaciones de la librería son automáticas e independientes de la aplicación de usuario.
- Google Play se convierte en una fuente confiable en la que poder obtener la librería en todo momento.

2.4 OpenCV en IOS

Desde la versión 2.4.2 OpenCV tiene una versión oficial para iOS. Para incorporar OpenCV a proyectos iOS, básicamente hay dos posibilidades [21]:

- Recompilar la librería para iOS. Es un proceso largo y propenso a errores, pero tiene la ventaja de poder disponer de las funciones más recientes incorporadas en las últimas versiones de OpenCV. Sin embargo, en la mayoría de los casos es suficiente con incorporar al proyecto la librería ya precompilada.
- La forma más sencilla de incorporar OpenCV precompilado en un proyecto es añadirlo como un marco de trabajo o framework. En un proyecto XCode, un marco de trabajo es un conjunto empaquetado que incluye librerías, ficheros.xib, ficheros cabecera, documentación, imágenes y otros recursos.

El marco de trabajo de la versión oficial iOS de OpenCV se puede descargar en <http://opencv.org/downloads.html>. Una de las mayores ventajas de los marcos de trabajo es que se puede incorporarlos a un proyecto fácilmente, basta con arrastrar la carpeta del marco de trabajo (opencv2.framework) sobre el grupo Frameworks dentro del Navegador de Proyecto de XCode [21] (ver Fig 10).

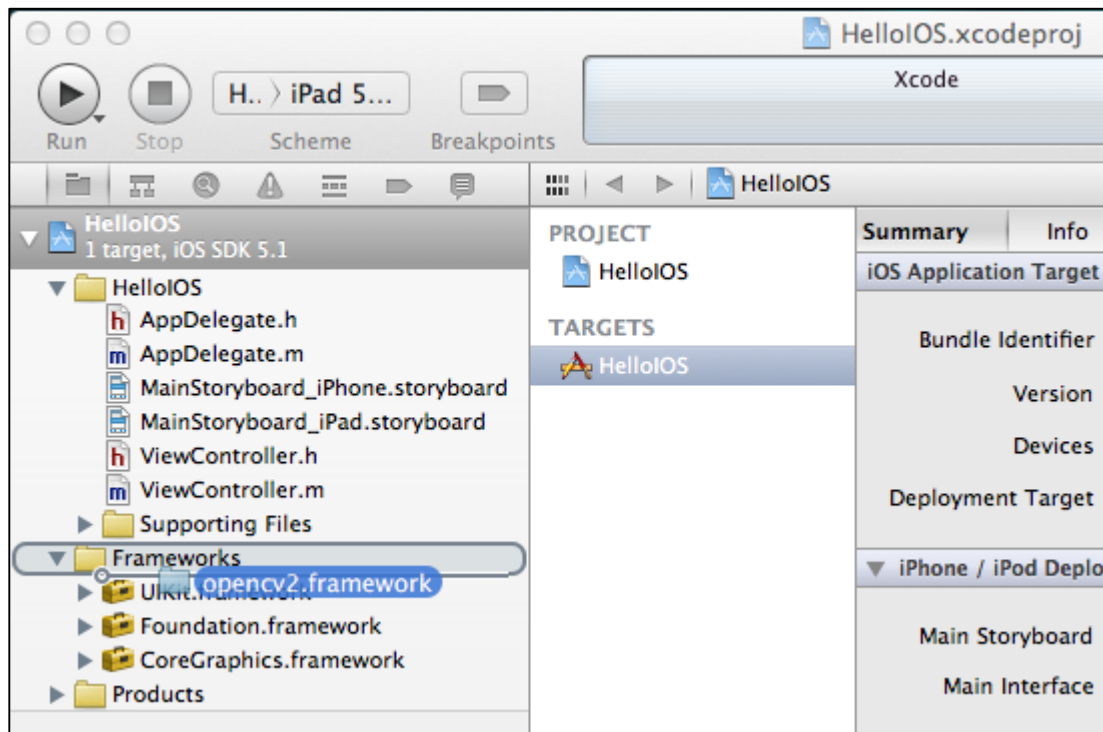


Figura 10. Adición del framework de OpenCV IOS en Xcode.

En ese momento se nos da la posibilidad de copiar los ficheros del marco de trabajo a la carpeta destino. Si estamos compartiendo el marco de trabajo entre múltiples proyectos tal vez querramos dejar esa opción desactivada [3].

Para los que quieran recompilar OpenCV para iOS, se recomienda seguir los pasos en http://docs.opencv.org/doc/tutorials/introduction/ios_install/ios_install.html#ios-installation. En el sitio se explica cómo obtener la última versión de OpenCV desde su repositorio Git y lanzar un script de compilación para iOS. El script genera un marco de trabajo, de nombre `opencv2.framework`.

Una vez se ha añadido el marco de trabajo, nuestro proyecto enlazará automáticamente con la librerías de OpenCV precompiladas. También están desde ese momento disponibles los ficheros cabecera, que debemos incluir en nuestros ficheros fuente (deben referenciarse con ruta relativa al marco de trabajo, p.ej. `#include <OpenCV/opencv/.../...>`)(ver Fig. 11).

```

1 //
2 // Prefix header for all source files of the 'VideoFilters' target in the 'VideoFilters' project
3 //
4
5 #import <Availability.h>
6
7 #ifndef __IPHONE_4_0
8 #warning "This project uses features only available in iOS SDK 4.0 and later."
9 #endif
10
11 #ifdef __cplusplus
12 #import <opencv2/opencv.hpp>
13 #endif
14
15 #ifdef __OBJC__
16     #import <UIKit/UIKit.h>
17     #import <Foundation/Foundation.h>
18 #endif

```

Figura 11. Cabeceras de trabajo en un fichero fuente IOS.

Para poder usar el marco de trabajo OpenCV en nuestro proyecto es necesario añadir otros marcos de trabajo de Apple. Se puede añadir estos marcos de trabajo al proyecto con la misma lista Link Binary with Libraries mencionada anteriormente. La tabla II muestra los marcos de trabajos necesarios y opcionales a añadir:

TABLA II. MARCOS DE TRABAJO NECESARIOS Y OPCIONALES PARA TRABAJAR CON OPENCV

Marcos de trabajo necesarios	AVFoundation.framework ImageIO.framework libz.dylib
Marcos de trabajo opcionales (necesarios para captura de vídeo)	CoreVideo.framework CoreMedia.framework

Ejemplo de detección de caras

OpenCV dispone de un detector de caras que es ampliamente utilizado por su velocidad y precisión. El ejemplo sigue la siguiente secuencia de pasos:

1. Cargar una imagen
2. Detectar una cara en ella
3. Si se encontró una cara, recuadrarla en la imagen
4. Salir del programa cuando se toque la pantalla

El código para la ejecución es el siguiente (Ver Fig.12):

```
1 - (void) viewDidLoad
2 {
3     [super viewDidLoad];
4
5     // Carga una imagen y la inserta en la vista actual...
6     UIImageView *myView = [[UIImageView alloc]
7         initWithFrame:self.view.frame];
8     UIImage *testImage = [UIImage imageNamed:@"Yo.jpg"];
9     myView.image = testImage;
10    [self.view addSubview:myView];
11
12    // Carga la cascada Haar de los recursos del proyecto...
13    cv::CascadeClassifier _faceCascade;
14    NSString *faceCascadePath = [[NSBundle mainBundle]
15        pathForResource:@"haarcascade_frontalface_alt2"
16        ofType:@"xml"];
17    if (!_faceCascade.load([faceCascadePath
18        cStringUsingEncoding:NSUTF8StringEncoding])) {
19        NSLog(@"No pude cargar la Haar_cascade: _
20            %@", faceCascadePath);
21    }
22
23    // Detectar caras...
24    std::vector<cv::Rect> faces;
25    cv::Mat mat;
26    [CVImageConverter CVMat:mat FromUIImage:testImage error:NULL];
27    _faceCascade.detectMultiScale(mat, faces, 1.1, 2,
28        CV_HAAR_FIND_BIGGEST_OBJECT | CV_HAAR_DO_ROUGH_SEARCH,
29        cv::Size(60, 60));
30
31    // Dibujar resultados...
32    if (faces.size())
33    {
34        // recuadrar la cara detectada...
35        cv::rectangle(mat, cv::Point(faces[0].x, faces[0].y),
36            cv::Point(faces[0].x+faces[0].width,
37                faces[0].y+faces[0].height), CV_RGB(255, 0, 0), 2);
38    }
39    else {
40        // escribir texto 'no face'
41        int fontFace = cv::FONT_HERSHEY_SIMPLEX;
42        double fontScale = 12;
43        cv::putText(mat, "no_face", cv::Point(10, mat.rows/2),
44            fontFace, fontScale, CV_RGB(255, 0, 0), 20);
45    }
46
47    // Mostrar imagen resultado...
48    myView.image = [CVImageConverter UIImageFromCVMat:mat
49        error:NULL];
50 }
```

Figura 12. Código para detección de caras.

El resultado del programa cuando se ejecuta en el simulador se muestra en la Fig. 13:



Figura 13. Resultado del ejemplo ejecutándose en el simulador

2.5 Técnicas de Reconocimiento facial

2.6 Eigenfaces

El Análisis de Componentes Principales (PCA) fue propuesta independientemente por Karl Pearson (1901) y Harold Hotelling (1933) a su vez un conjunto de variables correlacionadas, posiblemente, en un conjunto más pequeño de variables no correlacionadas. La idea es que un conjunto de datos de alta dimensión es a menudo descrito por variables correlacionadas, por lo que sólo unas pocas dimensiones significativas suponen la mayor parte de la información. El método PCA encuentra las direcciones con la mayor varianza en los datos, llamado componentes principales.

El principal problema de la representación de imágenes es su alta dimensionalidad. Imágenes en blanco y negro con dos dimensiones $p \times q$ ocupan un vector con un tamaño $m=p.q$. La cuestión es, ¿todos los datos de dichos vectores son necesarios? Solo se puede tomar una decisión cuando existe una variación en los datos. Por tanto, se busca la información más relevante.

El Análisis de Componente Principal (PCA en inglés) se encarga de que un conjunto de variables posiblemente correladas se conviertan en un conjunto de variables incorreladas [22]. Los métodos PCA encuentran la dirección con la mayor varianza en los datos, llamados componentes principales.

Descripción del algoritmo

Suponga $x = \{x_1, x_2, \dots, x_n\}$ un vector aleatorio con observaciones $x_i \in R^d$.

1. Calcule la media μ

$$\mu = \frac{1}{n} \sum_{i=1}^n x^i$$

2. Calcule la covarianza Matriz S

$$S = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)(x_i - \mu)^T$$

3. Calcule los valores λ_i y los autovalores v_i de S

$$Sv_i = \lambda_i v_i, i = 1, 2, \dots, n$$

4. Ordena los autovalores de forma descendiente por su autovalor. Los componentes principales k son los autovalores correspondientes a los mayores autovalores k.

Los componentes principales k del vector observado x son dados por:

$$y = W^T(x - \mu)$$

donde $W = (v_1, v_2, \dots, v_k)$

La reconstrucción desde las bases PCA se hace con:

$$x = Wy + \mu$$

donde $W = (v_1, v_2, \dots, v_k)$

Resumen

El EigenFace realiza la identificación facial:

- Proyectando todas las muestras a entrenar en el subespacio PCA.
- Proyectando la imagen a identificar en el subespacio PCA.

- Encontrando el vecino mas cercano entre la proyección de las imágenes entrenadas y la proyección de la imagen a identificar.

2.7 Fisherfaces

El PCA, que es la base del método EigenFace, encuentra combinaciones lineales de características que maximicen la varianza total en los datos [23]. Aunque es claramente una manera bastante poderosa para representar datos, no implica que no pueda perderse mucha información cuando se utiliza ese método.

Imagine una situación donde la varianza de los datos es generada por una fuente externa, como puede ser la iluminación. Los componentes identificados por un PCA no contiene ninguna información discriminativa, así que las muestras proyectadas se correlan juntas y la clasificación, por tanto, es imposible de realizar.

El Análisis Discriminatorio Lineal (LDA en inglés) consigue una reducción en la dimensionalidad de una especificación de una clase. Fue creado por el estadístico Sir R.A. Fisher [24]. Para poder encontrar la combinación de las características que diferencian mejor las clases, el LDA maximiza la proporción entre las clases y la dispersión de las clases, en vez de maximizar simplemente la dispersión total de las clases.

La idea es simple: una clase debe agrupar su proyección lo más pegada posible, mientras que las diferentes clases existentes deben estar tan alejadas unas de otras como sea posible en la representación de las proyecciones. Esto fue reconocido, también, por los investigadores Belhumeur, Hespanha y Kriegman, que aplicaron el LDA en reconocimiento facial.

Descripción del algoritmo.

Sea X un vector aleatorio con las muestras dibujadas de c clases:

$$X = \{X_1, X_2, \dots, X_c\}$$

$$X = \{X_1, X_2, \dots, X_n\}$$

Las matrices de dispersión S_B y S_W se calculan como:

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

$$S_W = \sum_{i=1}^c \sum_{x_j \in x_i} (x_j - \mu_i) (x_j - \mu_i)^T$$

donde μ es la media total:

$$\mu = \frac{1}{n} \sum_{i=1}^n x_n$$

y μ_i es la media de la clase $i \in \{1, 2, \dots, c\}$:

$$\mu_i = \frac{1}{|X_i|} \sum_{x_j \in x_i} x_j$$

El algoritmo clásico Fisher ahora busca, para una proyección W , maximizar el criterio de separación entre clases:

$$W_{opt} = \frac{\argmax_W W^T S_B W}{W^T S_W W}$$

Se obtiene una solución para esta optimización del problema, resuelto por el Problema de Autovalor General:

$$S_B V_i = \lambda_i S_W V_i$$

$$S_B V_i = \lambda_i S_W V_i$$

Queda un problema por resolver: El rango de S_W es al menos $(N-c)$, con N muestras y c clases. En problemas del patrón de reconocimiento, el número de muestras N es casi siempre menor que la dimensión de los datos de entrada (el número de píxeles), así que la matriz de dispersión S_W se convierte en singular. Se resuelve realizando un PCA en los datos proyectando las muestras en el espacio de dimensiones $(N-c)$. Un LDA puede realizarse a continuación para reducir los datos, ya que S_W ya no es singular.

La optimización del problema, por tanto, puede ser escrita como:

$$W_{pca} = \frac{\argmax_W |W^T S_T W|}{W}$$

$$W_{fld} = \frac{\argmax_W |W^T W_{pca}^T S_B W_{pca} W|}{W |W^T W_{pca}^T S_W W_{pca} W|}$$

La transformación de la matriz W , que proyecta la muestra en el espacio dimensional ($c-1$), es dada por:

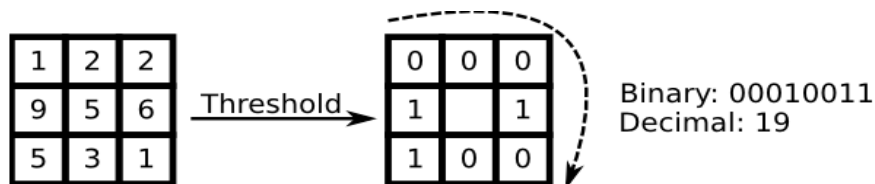
$$W = W_{fld}^T W_{pca}^T$$

2.8 Patrones Binarios Locales Histogramas (LBPH)

Los modelos EigenFace y FisherFace toman un enfoque holístico para la identificación fácil. La información es tratada como un vector con un espacio dimensional muy alto [25]. El enfoque realizado por EigenFace maximiza la dispersión total, el cual puede conllevar problemas si la varianza se debe por una fuente externa, ya que los componentes con una varianza máxima sobre todas las clases existentes no son necesariamente útiles para la clasificación. Para preservar alguna información discriminatoria, se aplica el LDA y se optimiza, como se describió para el modelo FisherFace.

Ahora, la idea no es mirar la imagen por completo como un vector con una gran dimensionalidad, sino describir solo las características locales de un objeto. Las características que se extraen de esta forma tendrán una baja dimensionalidad implícitamente. En esto se basa el operador de Patrones Binarios Locales (LBP en inglés). Se basa en el análisis de texturas en 2D.

La idea principal de LBP es resumir la estructura local de una imagen mediante la comparación de cada píxel con sus píxeles vecinos. Se toma un píxel como centro y se limita el valor de los vecinos. Es decir, si la intensidad del píxel central es mayor que su vecino, este se denota con un cero, y si, por el contrario, la intensidad del vecino es mayor o igual a la intensidad del vecino central, entonces se denota con un uno.



Descripción del algoritmo:

Una descripción más formal del operador LBP se puede dar como:

$$LBP(X_c, Y_c) = \sum_{p=0}^{P-1} 2^p S(i_p - i_c)$$

con (X_c, Y_c) como el píxel central con intensidad i_c ; i_p la intensidad del píxel vecino. s es la función signo definido como:

$$s(x) = \begin{cases} 1 & \text{si } x \geq 0 \\ 0 & \text{e.o.c.} \end{cases}$$

Esta descripción permite capturar detalles con mucha precisión en las imágenes [26]. Para un punto (X_c, Y_c) , la posición de su vecino (X_p, Y_p) , $p \in P$, puede ser calculado como:

$$x_p = x_c + R \cos\left(\frac{2\pi p}{P}\right)$$

$$y_p = y_c + R \sin\left(\frac{2\pi p}{P}\right)$$

donde R es el radio del círculo que forman los vecinos y P es el número de puntos de muestra.

El operador es una extensión de la codificación original LBP, por eso a veces es llamado LBP Extendido. Si los puntos coordenados en el círculo no corresponden a las coordenadas de la imagen, el punto debe ser interpolado. OpenCV implementa una interpolación bilineal:

$$f(x, y) \approx [1 - x] \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix} \begin{bmatrix} 1-y \\ y \end{bmatrix}$$

Por definición, el operador LBP es robusto frente a transformaciones en escala de grises monótonas. Por último, se incorpora la información espacial en el modelo de identificación facial. La representación propuesta es dividir la imagen LBP en regiones locales y extraer la información del histograma de cada una. El vector de características espaciales mejorado es obtenido mediante la concatenación de los histogramas locales (no una fusión). Estos histogramas son llamados como histogramas de Patrones Binarios Locales (LBPH en inglés).

CAPÍTULO III:

3. Herramientas de software y hardware seleccionadas

3.1 Herramientas de software

En este apartado hablaremos sobre todas las herramientas de software utilizadas para el desarrollo del presente proyecto de titulación.

3.1.1 Android

A finales de 2010 las cifras convirtieron a Android en la plataforma líder a nivel mundial en el mercado de los teléfonos inteligentes [27].

Una de las principales características de Android es que se trata de un proyecto de código abierto que puede utilizar cualquier fabricante de hardware e incluso personalizar a sus necesidades. Esta cualidad de Android permite que exista compatibilidad de las aplicaciones con los dispositivos certificados en el programa de compatibilidad de Android.

Es por este motivo que ha sido seleccionado para el desarrollo del presente proyecto dada su popularidad, además de que el software de visión artificial a utilizarse tiene limitada compatibilidad de plataformas móviles.

3.1.2 SQLite

Es una biblioteca que implementa un motor autónomo, sin servidor, sin configuración de base de datos SQL. El código para SQLite es de dominio público y por lo tanto libre para uso para cualquier propósito, comercial o privado [28].

SQLite se encuentra actualmente en más aplicaciones que podemos contar, incluyendo varios proyectos de alto perfil. Es un motor de base de datos SQL incorporado. A diferencia de la mayoría de las otras bases de datos SQL. SQLite no tiene un proceso de servidor independiente.

Lee y escribe directamente en archivos de disco ordinarios. Una base de datos completa de SQL con varias tablas, índices, triggers y vistas, está contenida en un archivo de disco único. El formato de archivo de base de datos es multiplataforma se puede copiar libremente una base de datos entre sistemas de 32 bits y de 64 bits.

SQLite es una opción popular motor de base de datos en memoria limitada gadgets como teléfonos móviles, PDAs y reproductores MP3. Hay un equilibrio entre el uso de memoria y velocidad. SQLite se ejecuta generalmente más rápido, mientras más memoria que le de, pero sin embargo, el rendimiento suele ser bastante bueno, incluso en entornos de poca memoria.

La base de código SQLite es apoyado por un equipo internacional de desarrolladores que trabajan en SQLite a tiempo completo. Los desarrolladores siguen ampliando las capacidades de SQLite y mejorando su fiabilidad y rendimiento al tiempo que mantiene la compatibilidad con la especificación publicada interfaz, sintáxis SQL, y formato de archivo de base de datos.

El código fuente es totalmente gratuito para todo aquel que quiera, pero el apoyo profesional también está disponible.

3.1.3 OpenCV Android

OpenCV (Open Source Computer Vision Library) es una fuente abierta de visión por computador y una biblioteca de software de aprendizaje automático [29].

OpenCV fue construido para proporcionar una infraestructura común para aplicaciones de visión por ordenador y para acelerar el uso de la percepción de la máquina en los productos comerciales.

Al ser un producto de licencia BSD, OpenCV hace que sea fácil para las empresas el utilizar y modificar el código. La biblioteca cuenta con más de 2.500 algoritmos optimizados, que incluye un amplio conjunto de algoritmos de visión por computador y aprendizaje automático con tecnología de última generación.

Estos algoritmos se pueden utilizar para:

- Detectar y reconocer rostros,
- Identificar objetos.
- Clasificar las acciones humanas en videos, movimientos de cámara, objetos pista pista en movimiento
- Extraer modelos 3D de objetos, producen nubes de puntos 3D de cámaras estéreo.
- Encontrar imágenes similares de una base de datos de imágenes.
- Eliminar los ojos rojos de las imágenes tomadas con flash.
- Seguir los movimientos de los ojos,
- Reconocer paisajes y establecer marcadores para cubrir realidad aumentada, etc.

La biblioteca se utiliza ampliamente en las empresas, grupos de investigación y de los organismos gubernamentales.

Junto con empresas bien establecidas como Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota, que emplean la biblioteca, hay muchas nuevas empresas como Applied Minds, VideoSurf y Zeitera, que hacen un amplio uso de OpenCV [29].

Cuenta con interfaces de C ++, C, Python, Java y MATLAB y es compatible con Windows, Linux, Android y Mac OS. OpenCV se inclina principalmente hacia las aplicaciones de visión en tiempo real y se aprovecha de instrucciones MMX y SSE cuando esté disponible.

OpenCV está escrita de forma nativa en C ++ y tiene una interfaz de plantilla que funciona a la perfección con contenedores STL.

3.1.4 NDK Android.

El NDK es un conjunto de herramientas que le permite implementar partes de una aplicación utilizando lenguajes de código nativo, como C y C ++. Por lo general, los buenos casos de uso para el NDK son aplicaciones que hacen uso intensivo de la CPU, como motores de juego, procesamiento de señales, y simulación de la física. [30]

Hay que entender que el NDK no beneficia a la mayoría de las aplicaciones. En general, sólo se debe utilizar el NDK si es esencial para su aplicación, nunca porque simplemente se prefiera programar en C / C ++.

El desarrollo de aplicaciones para Android se realiza habitualmente en lenguaje Java. Sin embargo, con este lenguaje también es posible emplear librerías programadas en C/C++. Para conseguirlo es necesario compilar dichas librerías para la plataforma ARM de destino empleada en los sistemas móviles Android y emplear la interfaz JNI (Java Native Interface) para poder acceder a ellas desde la máquina virtual Java (en concreto desde la DVM).

Este proceso se lleva a cabo con las herramientas proporcionadas en el NDK (Native Development Kit) para la plataforma anfitrión empleada para el desarrollo de las aplicaciones Android. El uso de código nativo C/C++ puede ser especialmente ventajoso en algunas aplicaciones Android donde se requiera el empleo de librerías específicas escritas en dichos lenguajes.

Sin embargo, por regla general para la programación de aplicaciones Android es suficiente con el empleo de las API proporcionadas en el SDK. Antes de usar NDK es preciso sopesar los beneficios obtenidos, generalmente relativos a desempeño, frente a la dificultad añadida al proceso de desarrollo.

Las aplicaciones de procesamiento de imagen en plataformas móviles pueden aprovechar las librerías como OpenCV con una dilatada trayectoria en el desarrollo de aplicaciones clásicas empleando NDK para aquellos algoritmos con mayores requerimientos de desempeño o que aún no tienen una API en las nuevas plataformas [30].

3.1.5 Eclipse IDE

Eclipse es una plataforma de desarrollo de código abierto basada en Java. Por si misma, es simplemente un marco de trabajo y un conjunto de servicios para la construcción del entorno de desarrollo de los componentes de entrada [31].

Afortunadamente, Eclipse tiene un conjunto de complementos, incluidas las Herramientas de Desarrollo de Java (JDT). Eclipse también incluye el Entorno de Desarrollo de

Complementos (PDE), que es de interés principalmente para los desarrolladores que quieren extender Eclipse, dado que les permite construir herramientas que se integran sin dificultades con el entorno de Eclipse.

Dado que todo en Eclipse es un complemento, todos los desarrolladores de herramientas tienen un campo de juego de nivel para ofrecer extensiones a Eclipse y para proporcionar un entorno de desarrollo integrado y unificado para los usuarios.

Esta paridad y consistencia no está limitada a las herramientas de desarrollo de Java. Aunque Eclipse se escribe en el lenguaje Java, su uso no se limita al lenguaje Java. Por ejemplo, los complementos se encuentran disponibles o planificados para incluir soporte para los lenguajes de programación como C/C++ y COBOL. El marco de trabajo de Eclipse puede también utilizarse como base para otros tipos de aplicaciones que no se relacionen con el desarrollo del software, como los sistemas de gestión de contenido. En la Fig. 14 podemos observar la GUI Eclipse [31]:

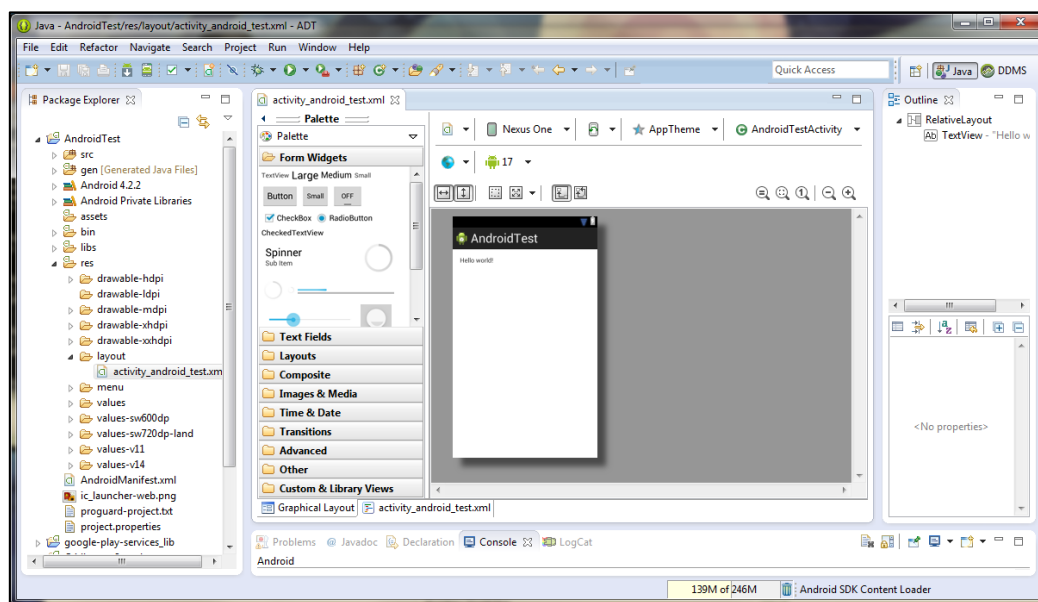


Figura 14. Interfaz de usuario de Eclipse Android.

3.1.6 Fritzing

Fritzing es una iniciativa de hardware de código abierto que hace que la electrónica sea accesible como material creativo para cualquier persona [32].

Es una herramienta de software, un sitio web de la comunidad y los servicios para el procesamiento y Arduino, que fomenta un ecosistema creativo que permite a los usuarios documentar sus prototipos, comparten con los demás, enseñar electrónica en un salón de clases, y el diseño y fabricación de PCB profesionales, en la Fig. 15 podemos observar la interfaz de usuario de Fritzing:

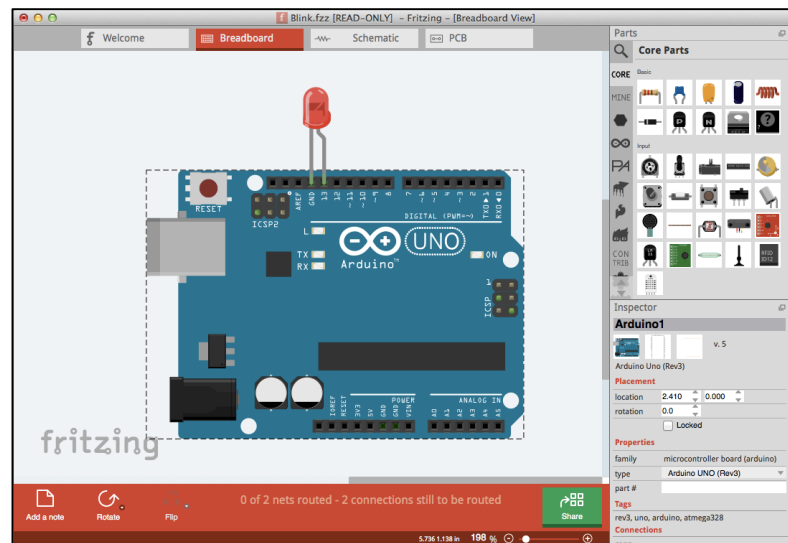


Figura 15. Interfaz de usuario de Fritzing

3.1.7 Arduino software

El Software Arduino (IDE) de código abierto ayuda a que sea fácil de escribir código y subirlo a la placa. Se ejecuta en Windows, Mac OS X y Linux. El entorno está escrito en Java y basa su procesamiento en otro software de código abierto [33].

Este software se puede utilizar con cualquier placa Arduino. En la Fig. 16 podemos apreciar la interfaz de usuario de Arduino software.

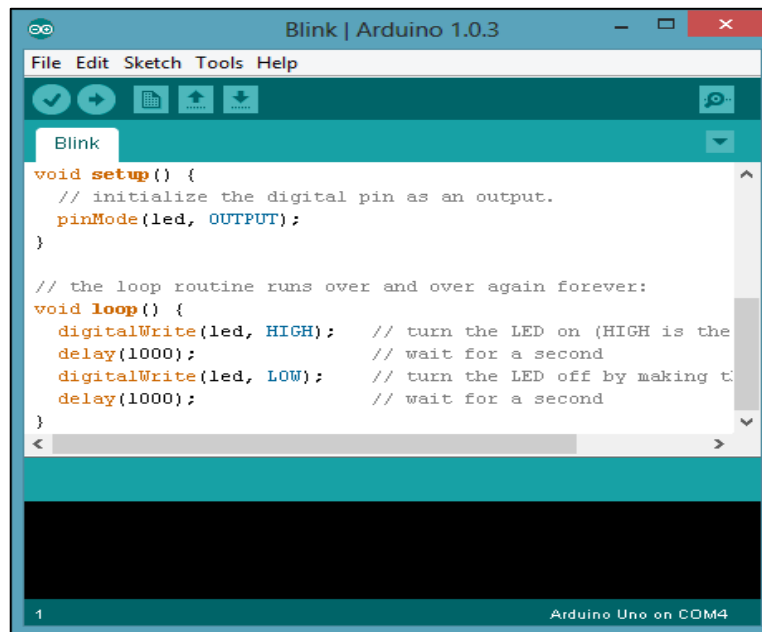


Figura 16. Interfaz de usuario de Arduino

3.2 Herramientas de hardware.

En este apartado se describirá las diferentes herramientas de hardware que se utilizaron para la implementación del dispositivo electrónico que actuará en el sistema de encendido del automóvil.

3.2.1 Arduino

Arduino es una plataforma electrónica de código abierto basado en hardware y software fácil de usar. Está dirigido a cualquier persona que hace proyectos interactivos. El corazón de Arduino es un microcontrolador. En realidad, el resto de la placa se ocupa de facilitar la alimentación y permitir que se comunique con el ordenador al que está conectado [34].

Arduino es un pequeño ordenador en un chip. Tiene todo y más de lo que contenían los primeros ordenadores domésticos. Dispone de un procesador, memoria RAM (memoria de acceso aleatorio) para contener datos, unos cuantos kilobytes de memoria EPROM (ROM programable borrrable) o de memoria Flash para contener nuestros programas, y tiene pines de entrada y salida.

Las entradas pueden leer señales digitales (¿el interruptor está abierto o cerrado?) y analógicas (¿cuál es la tensión en un pin?). Esto nos permite conectar innumerables tipos de sensores de luz, temperatura, sonido, etc.

Las salidas también pueden ser analógicas o digitales. Así, se puede establecer que un pin esté activado o desactivado (5 V o 0 V) y esto puede encender o apagar los LED directamente, o se puede usar la salida para controlar dispositivos más potentes, como motores.

También pueden proporcionar tensión de salida analógica. Es decir, se puede fijar la salida de un pin a una determinada tensión, lo que permite controlar la velocidad de un motor o el brillo de una bombilla, por ejemplo, en lugar de solo encenderlo o apagarlo, en la Fig. 17 podemos observar una placa básica de Arduino uno:



Figura 17. Placa de Arduino uno.

3.2.2 Arduino Mega 2560

El Arduino Mega 2560 es una placa electrónica basada en el microcontrolador Atmega2560. Cuenta con 54 pines digitales de entrada / salida de los cuales 15 se pueden utilizar como salidas PWM, 16 entradas analógicas, 4 UARTs (puertas seriales), un

oscilador de cristal 16 MHz, una conexión USB, un conector de alimentación, una cabecera ICSP, y un botón de reinicio [33].

Contiene todo lo necesario para apoyar el microcontrolador; basta con conectarlo a un ordenador con un cable USB o el poder con un adaptador de CA o la batería a CC para empezar.

El Mega2560 difiere de todas las placas anteriores en que no utiliza el chip controlador de USB a serial FTDI. En lugar de ello, cuenta con la ATmega16U2 programado como convertidor USB a serie.

Se determinó el uso de este Arduino por las capacidades y mayor número de puertos de conexión necesarios para la integración de más partes necesarias para la construcción del dispositivo electrónico.

En la Fig. 18 podemos observar la placa de Arduino Mega 2560 y en la Tabla III tenemos las especificaciones técnicas:

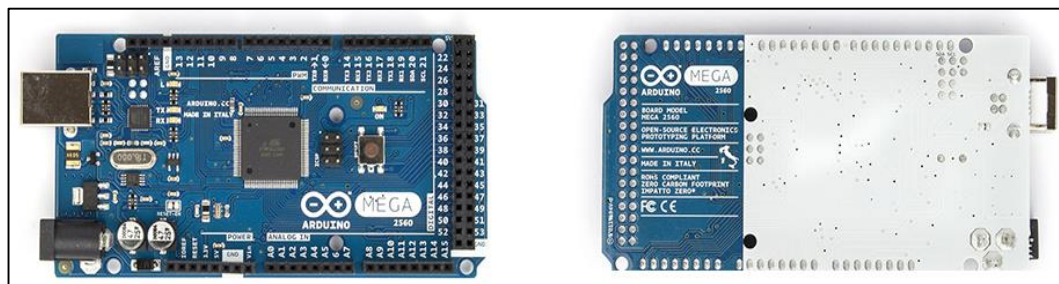


Figura 18. Arduino Mega 2560, vista delantera y posterior.

TABLA III. ESPECIFICACIONES TÉCNICAS ARDUINO MEGA 2560

Microcontroladores	Atmega2560
Tensión de funcionamiento	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límites)	6-20V
Digital pines I / O	54 (de las cuales 15 proporcionan salida PWM)
Pines de entrada analógica	16
Corriente DC por Pin I / O	40 mA
Corriente DC de 3.3V Pin	50 mA
Memoria Flash	256 KB de los cuales 8 KB utilizado por el gestor de arranque
SRAM	8 KB
EEPROM	4 KB
Velocidad del reloj	16 MHz

Memoria

El Atmega2560 tiene 256 KB de memoria flash para el almacenamiento de código de los cuales 8 KB se utiliza para el gestor de arranque, 8 KB de SRAM y 4 KB de EEPROM.

Entrada y Salida

Cada uno de los 54 pines digitales en el Mega se puede utilizar como una entrada o salida, utilizando `pinMode ()`, `digitalWrite ()`, y `digitalRead ()` funciones. Funcionan a 5 voltios.

Cada pin puede proporcionar o recibir un máximo de 40 mA y tiene una resistencia de pull-up (desconectado por defecto) de 20 a 50 kOhm. Además, algunos pines tienen funciones especializadas:

Serial: 0 (RX) y 1 (TX); Serie 1: 19 (RX) y 18 (TX); Serial 2: 17 (RX) y 16 (TX); Serial 3: 15 (RX) y 14 (TX). Se utiliza para recibir (RX) y transmitir datos en serie (TX) TTL. Pines 0 y 1 están también conectados a los pines correspondientes del ATmega16U2 USB-to-TTL chip de serie.

PWM: 2 a 13 y 44 a 46: Proporciona salida PWM de 8 bits con la función analogWrite ().

SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS): Estos pines admite la comunicación SPI utilizando la librería SPI.

LED 13: LED incorporado conectado al pin digital 13. Cuando el pasador es de alto valor, el LED está encendido, cuando el pasador es bajo, es apagado.

3.2.3 Módulo Bluetooth HC-06

El módulo Bluetooth de serie HC, consiste en un módulo de interfaz serie Bluetooth y adaptador Bluetooth Los módulos de interfaz serial bluetooth que existen son [35]:

- Nivel Industrial: HC-03, HC-04(HC-04-M, HC-04-S)
- Nivel Civil level: HC-05, HC-06(HC-06-M, HC-06-S) HC-05-D, HC-06-D.

El módulo de serie Bluetooth se utiliza para la conversión de puerto serie a Bluetooth. Estos módulos tienen dos modos: maestro y esclavos. El dispositivo que lleva el nombre de número par se define como maestro o esclavista cuando fuere de la fábrica y no puede ser cambiado al otro modo.

En la Fig. 19 podemos observar el módulo hc-06:

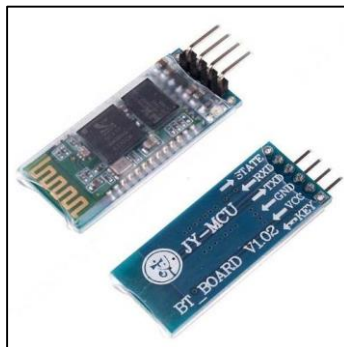


Figura 19. Módulo Bluetooth hc-06.

3.2.4 Módulo Lector SD

El módulo Tarjeta SD sirve para leer y escribir los datos a través de Arduino. Puede utilizarse para reproductor de MP3, control de MCU sistema / ARM [36].

Características:

- De fácil interconexión con Arduino y otros Microcontroladores
- Ideal para proyectos Arduino MP3 Player
- Pines SD SPI: MOSI, SCK, MISO y CS.
- Soporta entradas de 5V/3.3V.

En la Fig. 20 podemos apreciar el lector de SD descrito anteriormente:

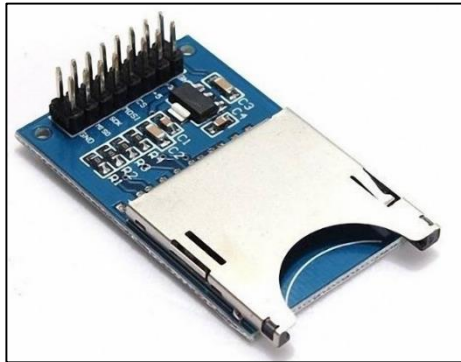


Figura 20. Módulo lector de SD.

3.2.5 Módulo Relay o Relé

Esta placa está diseñada para interactuar con dispositivos que necesiten ser accionados [37].

Características

- Voltaje de Alimentación: 5V.
- Corriente soportada: 10 A máximo.
- Corriente de control: 15 mA.
- Voltaje de control: 110/120 V.

La Fig. 21 muestra la placa relé de control:



Figura 21. Placa relé de control.

CAPÍTULO IV:

e. MATERIALES Y MÉTODOS

El presente proyecto se realizó utilizando diferentes métodos y técnicas para alcanzar los objetivos propuestos:

1. Métodos

- **Método Científico:**

Este método se utilizó para la propuesta, observación, construcción y pruebas del prototipo obteniendo los resultados del funcionamiento y diseño del prototipo.

- **Método Deductivo:**

Este método permitió obtener la información global necesaria correspondiente al hardware y software; así poder aplicarlos para construir el prototipo propuesto.

- **Método Inductivo:**

A través de este método se plantean los principales problemas en seguridad en un automóvil para determinar si la solución planteada permite resolverlos.

- **Metodología Mobile-D**

Para el desarrollo de la aplicación móvil se utilizará la metodología de desarrollo de aplicaciones móviles Mobile-D comprendida en 5 fases [38]:

- **Exploración:** En esta fase se establecerá las características de la aplicación móvil. Esto se realizará en tres etapas: establecimiento actores, definición del alcance y el establecimiento de proyectos.
- **Iniciación:** Lo que se realizará en esta fase será preparar e identificar todos los recursos necesarios: recursos físicos, tecnológicos y de comunicaciones. Esta fase se divide en cuatro etapas: la puesta en marcha del proyecto, la planificación inicial, el día de prueba y día de salida.

- **Producción:** En esta fase ya se comienza a codificar la aplicación móvil, se repite la programación de tres días (planificación, trabajo, liberación) se repetirá hasta implementar todas las funcionalidades.
- **Estabilización:** Aquí se llevarán a cabo las últimas pruebas de integración para asegurar que la aplicación móvil completa funciona correctamente y también se realizará la producción de documentación.
- **Prueba y Reparación del sistema:** En esta fase ya se tendrá una versión estable y funcional de la aplicación. El producto terminado se probará y se eliminarán todos los defectos encontrados.

2. Técnicas

- **Observación directa:**

Esta técnica permitirá observar el mecanismo de encendido del automóvil para realizar la instalación del prototipo a desarrollarse.

- **Revisión Bibliográfica:**

Mediante esta técnica se buscará el sustento teórico para el desarrollo del proyecto en base a consultas de libros, artículos científicos, revistas científicas, páginas web confiables entre otros.

f. RESULTADOS

A continuación se describen los resultados de cada uno de los objetivos planteados para el desarrollo del presente proyecto de titulación, para lo que se ha utilizado herramientas, materiales, métodos, etc, elementos que han sido de vital importancia para el logro del proyecto antes mencionado.

2.9 ANÁLISIS

Con esta fase se busca realizar el análisis de la problemática y obtener todos los requerimientos del proyecto a desarrollar.

1.1 Exploración

En esta etapa se definen los requerimientos, y alcance del proyecto, bases fundamentales para el adecuado desarrollo de la aplicación móvil.

1.1.1 Establecimiento de los Grupos de Interés o Stakeholders

Los grupos de personas interesadas en la realización del presente proyecto son los siguientes:

- **Desarrollador:** Es la persona encargada del análisis, diseño, desarrollo y pruebas de la aplicación, en este caso el tesista.
- **Propietarios de autos:** Son todas las personas que poseen un vehículo y un dispositivo móvil con Sistema Operativo Android.

1.1.2 Requerimientos Iniciales

La aplicación móvil estará en capacidad de funcionar como un dispositivo de seguridad de acceso a un vehículo mediante el uso de datos personales del usuario, siendo en este caso el dueño del vehículo, los cuales se almacenarán dentro del dispositivo electrónico arduino.

1.1.2.1 Requerimientos Funcionales

Los requerimientos que la aplicación móvil permitirá se encuentran en la tabla IV:
La aplicación móvil debe permitir:

TABLA IV. REQUERIMIENTOS FUNCIONALES.

Código	Descripción	Prioridad
RF001	Al usuario detectar los dispositivos Bluetooth.	ALTA
RF002	Al usuario conectarse al dispositivo electrónico Arduino una vez que se haya terminado el proceso de detección.	ALTA
RF003	Al usuario verificar si el dispositivo se encuentra configurado o si el dispositivo se encuentra en estado funcional.	ALTA
RF004	Al usuario poder iniciar con el entrenamiento de su patrón facial a través de las cámaras que posea el dispositivo móvil para su posterior identificación.	ALTA
RF005	Al usuario guardar los datos del patrón facial obtenidos en el entrenamiento dentro del dispositivo Android.	ALTA
RF006	Al usuario obtener y almacenar los datos personales del usuario dentro del dispositivo electrónico Arduino.	ALTA
RF007	Al usuario guardar el nombre y la dirección MAC del dispositivo dentro del dispositivo electrónico Arduino.	ALTA
RF008	Al usuario administrar los dispositivos Bluetooth almacenados dentro del dispositivo electrónico Arduino.	MEDIA
RF009	Al usuario administrar sus datos personales almacenados dentro del dispositivo electrónico Arduino.	MEDIA
RF010	Al usuario poder usar cualquiera de las cámaras que posea el dispositivo móvil.	MEDIA
RF011	Al usuario poder detectar su rostro al enfocar su cámara.	ALTA
RF012	Al usuario verificar su identidad con los datos previamente guardados en el entrenamiento dentro del dispositivo electrónico Arduino.	ALTA
RF013	Al usuario desbloquear el dispositivo electrónico Arduino, aún cuando no existan las condiciones de luz	ALTA

	necesarias para el reconocimiento facial mediante otro mecanismo seguro.	
RF014	Al usuario administrar los rostros almacenados dentro del dispositivo móvil.	MEDIA
RF015	Al usuario permitir y denegar las conexiones entrantes hacia el dispositivo electrónico Arduino.	ALTA
RF016	Al usuario permitir reestablecer las configuraciones iniciales de la aplicación móvil y del dispositivo electrónico en caso de que así lo requiera.	MEDIA

1.1.2.2 Requerimientos no funcionales.

Los requerimientos no funcionales para el módulo para la aplicación móvil son descritos en la tabla V.

TABLA V. REQUERIMIENTOS NO FUNCIONALES.

Código	Descripción
RNF01	La aplicación móvil será desarrollada bajo la plataforma Android.
RNF02	La aplicación móvil posee una arquitectura cliente-servidor.
RNF03	La aplicación móvil utilizará una interfaz amigable.
RNF04	El Sistema será desarrollado bajo la plataforma de programación Java
RNF05	La aplicación móvil será multiusuario.
RNF06	El sistema utilizará una interfaz amigable.
RNF07	La conexión, procesamiento y transmisión de datos entre la aplicación móvil y el dispositivo electrónico Arduino debe realizarse en menos de 3 segundos.

1.1.2.3 Análisis de los Requerimientos

En base a los requerimientos establecidos se ha podido determinar los procesos a realizar en la tabla VI.

TABLA VI. MÓDULOS Y PROCESOS DE LA APLICACIÓN.

Módulo	Código	Proceso	Requerimientos
Módulo de Configuración inicial	P001	Detectar dispositivos bluetooth	RF001, RF002, RF003
	P002	Guardar capturas de rostro del usuario	RF004, RF005, RF010, RF011
	P003	Guardar datos personales de usuario	RF006
	P004	Guardar datos del dispositivo móvil Android	RF007
Módulo de administración	P005	Administrar los dispositivos Bluetooth	RF008
	P006	Administrar sus datos personales	RF009
	P007	Administrar rostros de usuarios	RF014
Módulo de entrenamiento y reconocimiento facial.	P008	Usar cámaras	RF010, RF011
	P009	Verificar identidad de usuario	RF012, RF013
Módulo de configuración de la aplicación móvil.	P010	Permitir y denegar conexiones entrantes	RF015
	P011	Reestablecer configuraciones de fábrica	RF016

1.1.3 Definición del Alcance

El alcance del presente proyecto se ha determinado a través de su categoría, limitaciones, los supuestos y dependencias.

1.1.3.1 Establecimiento de Categoría

La aplicación móvil es de tipo empresarial, puesto que va orientada hacia el sector automovilístico como una mejora en la seguridad y acceso a dichos autos.

1.1.3.2 Limitaciones

Las limitaciones de la aplicación son:

- La aplicación móvil sólo puede ser ejecutada en dispositivos con plataforma Android.
- La ejecución de la aplicación requiere que el dispositivo móvil tenga instalado OpenCV Manager.
- El dispositivo móvil debe tener conectividad Bluetooth.
- La aplicación móvil solo puede administrar un dispositivo electrónico Arduino a la vez.

1.1.3.3 Supuestos y dependencias

Los supuestos y dependencias de la aplicación son:

- La aplicación móvil recoge los datos de reconocimiento facial del dispositivo móvil.
- La aplicación móvil recoge los datos personales y de dispositivos móviles desde la memoria del dispositivo electrónico Arduino.

1.1.4 Establecimiento del Proyecto

En esta etapa se determinan los recursos físicos y técnicos necesarios para el desarrollo del proyecto. Las herramientas a utilizadas son las siguientes (ver tabla VII y tabla VIII):

TABLA VII. HERRAMIENTAS SOFTWARE

Nombre	Descripción
Eclipse Android	Eclipse Android brinda un completo entorno de desarrollo para la codificación de aplicaciones Android al estar compuesto por el IDE Eclipse y el sdk de Android.
Fritzing	Es un software de diseño electrónico que brinda las herramientas para diseño y simulación electrónica con licencia libre.
Arduino IDE	Es un entorno de desarrollo para la escritura y carga de código hacia la placa de Arduino, su licencia es libre.

OpenCV Android	Es una biblioteca de Visión Artificial la cual permite múltiples funciones como reconocimiento de objetos, detectar movimiento, detectar rostros, detectar partes del cuerpo, reconocer rostros la cual también cuenta con una licencia libre.
----------------	--

TABLA VIII. HERRAMIENTAS HARDWARE.

Nombre	Descripción
Sony Xperia M:	Es un teléfono móvil Android de gama media el cual se ha utilizado para poder ejecutar la aplicación móvil.
Samsung Galaxy TAB 4:	Es un tablet 3G de gama media que también se ha utilizado para probar y ejecutar la aplicación móvil desarrollada.
Arduino Mega 2560:	Es una placa electrónica perteneciente al proyecto Arduino que es una plataforma de software y hardware que simplifica el uso de la electrónica.
Módulo Bluetooth HC-06:	Es un dispositivo bluetooth que permite enviar y recibir datos sin necesidad de cables.
Módulo Lector SD:	Es un dispositivo electrónico que permite leer y escribir datos en un tarjeta SD.
Tarjeta SD:	Es un dispositivo de almacenamiento de capacidad variable.
Módulo Relay:	Es una placa electrónica diseñada para el corte de conexión eléctrica de un circuito.
Carro con Radio Control:	Es un auto de juguete pequeño elegido para la instalación del dispositivo electrónico Arduino.

2. DISEÑO

Con el diseño lo que se desea obtener son los diagramas que representen el funcionamiento de la aplicación, los procesos que va a ejecutar con el fin de cumplir los requerimientos definidos en la primera fase.

2.1 Inicialización

En esta etapa se realizan las actividades relacionadas a la configuración del ambiente de desarrollo y el diseño de la aplicación.

2.1.1 Configuración del Ambiente de Desarrollo

En esta actividad se realizó la configuración en el IDE Eclipse Android con el fin de que se pueda utilizar para el desarrollo de aplicaciones con visión artificial con OpenCV y el NDK de Android.

Tipo de Proyecto: Android Application Project. En este tipo de proyectos hay diferencias con los proyectos Android normales, ya que además del código Java que utiliza normalmente también se utiliza código nativo c y c++ con el fin de aprovechar los recursos de procesamiento de los procesadores móviles.

Configuraciones: En esta parte se ha hecho la inclusión de las librerías OpenCV para el uso de las funciones detección y reconocimiento de rostros, así como también la integración del NDK Android en el IDE Eclipse para funciones de compilación y ejecución, la configuración realizada se detalla a continuación:

1. Con el IDE ya instalado descargamos el paquete de OpenCV Android y el NDK Android
2. Una vez descargados procedemos a descomprimirlos en el directorio donde se lo quiera colocar (ver Fig. 22).

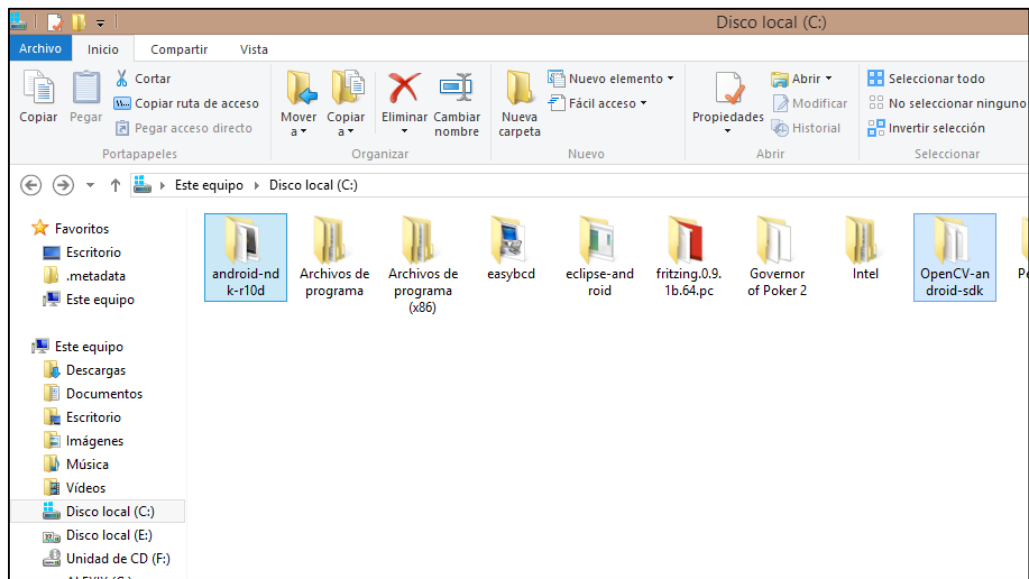


Figura 22. Directorio de instalación del NDK Android y OpenCV Android.

3. Abrimos Eclipse y seleccionamos como workspace el directorio de OpenCV (ver Fig. 23).

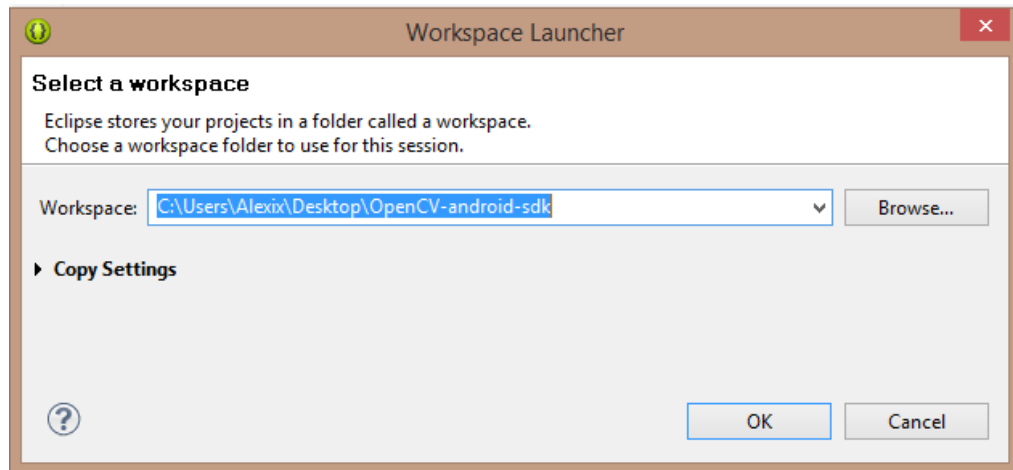


Figura 23. Selección de Directorio de trabajo.

4. Para la ejecución de las aplicaciones con OpenCV necesitamos tener configurado el NDK de Android para lo cual vamos a la barra de Menú de Eclipse y seleccionamos Window -> Preferences -> Android -> NDK y seleccionamos el directorio donde colocamos el NDK Android (ver Fig. 24).

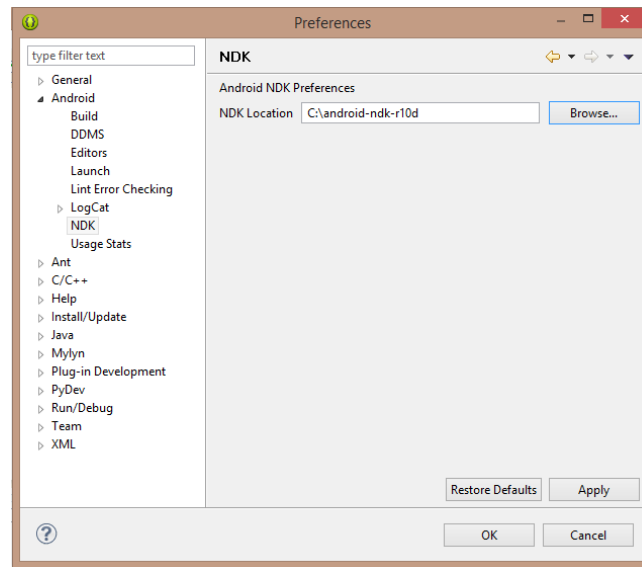


Figura 24. Selección del Directorio del NDK Android.

5. De la misma manera colocamos el directorio del NDK como variable de entorno (ver Fig. 25)

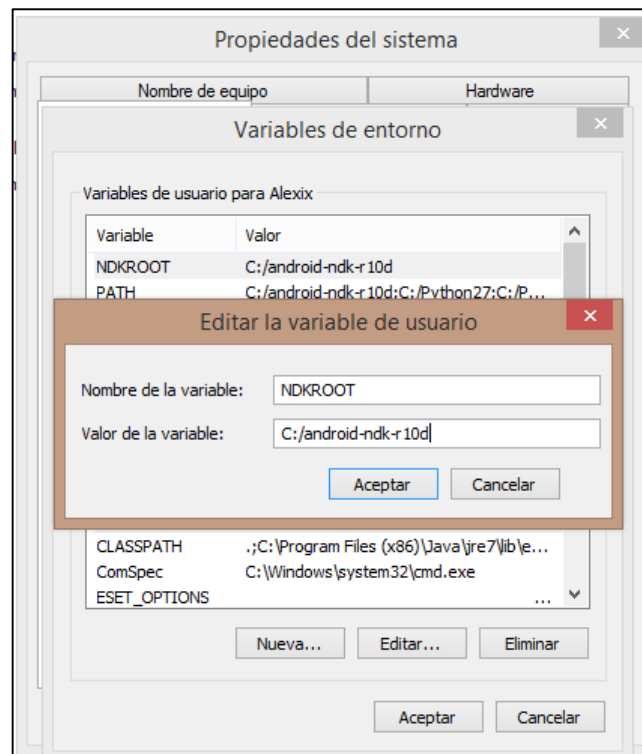


Figura 25. Configuración de variable de entorno.

6. Ahora procedemos a crear un nuevo proyecto Android normalmente (ver Fig. 26).

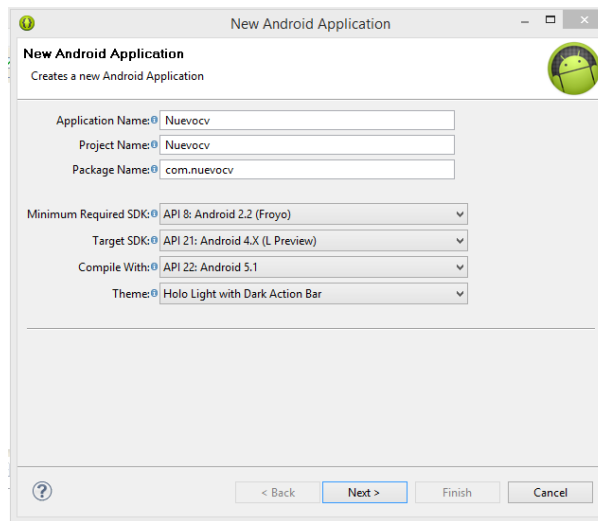


Figura 26. Creación de un nuevo Proyecto Android.

7. Luego de haber creado el proyecto hacemos para poder trabajar con OpenCV y el NDK de Android debemos convertir Android a un proyecto C/C++ para lo cual hacemos click derecho en el proyecto Android y vamos a New -> Other y nos aparecerá la ventana con diferentes opciones de proyecto, buscamos la opción C/C++ y seleccionamos Convert to a C/C++ Project (ver Fig. 27).

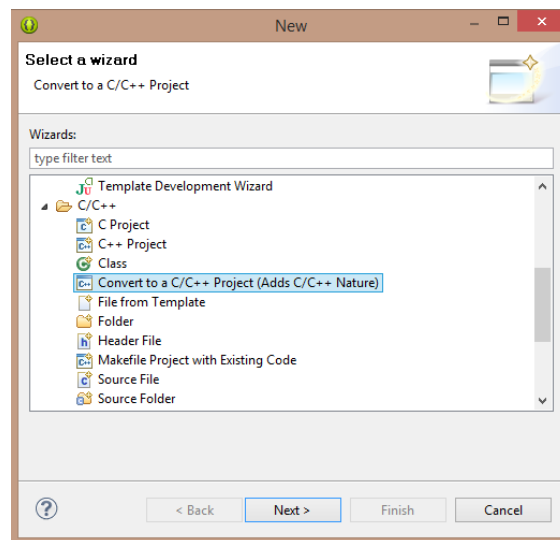


Figura 27. Conversión de proyecto a C/C++.

8. Luego seleccionamos el proyecto que queremos convertir y presionamos en Finish (ver Fig. 28).

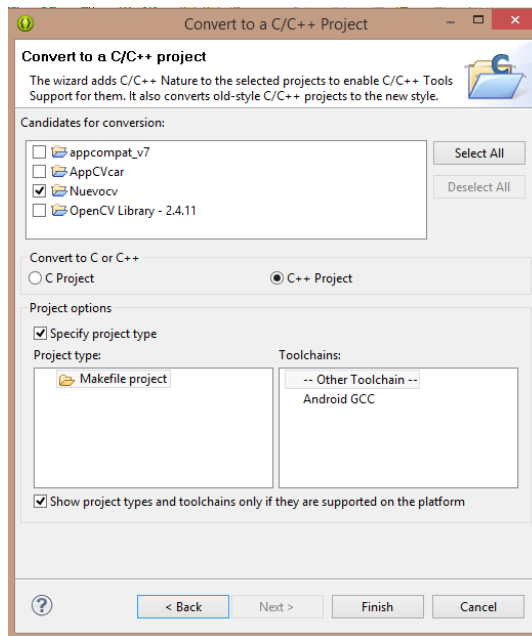


Figura 28. Final de Conversión de proyecto a C/C++.

9. Ahora sobre el mismo proyecto hacemos click derecho y seleccionamos Properties y Luego en Android, aquí agregamos las librerías de OpenCV (ver Fig. 29).

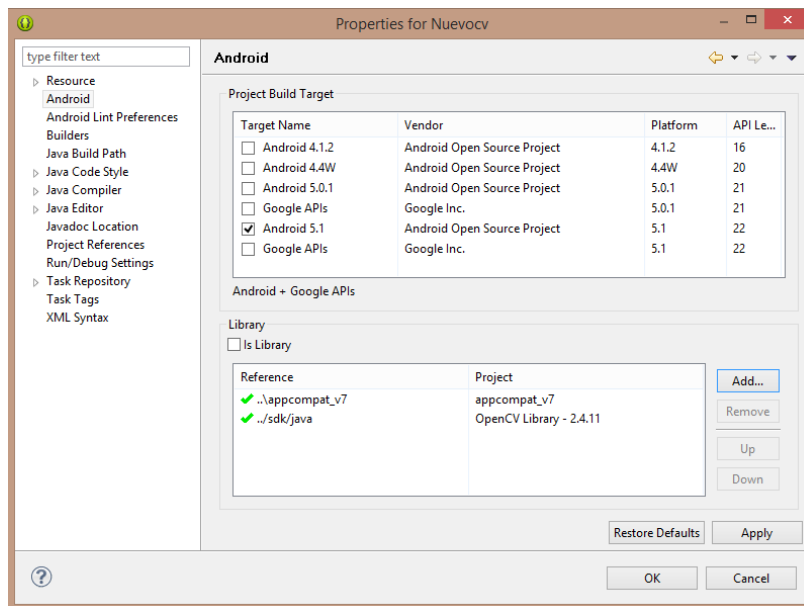


Figura 29. Agregar librerías de OpenCV.

10. Seleccionamos la opción C/C++ Build y escribimos en el campo Build command lo siguiente `${NDKROOT}/make-build.cmd` (ver Fig. 30).

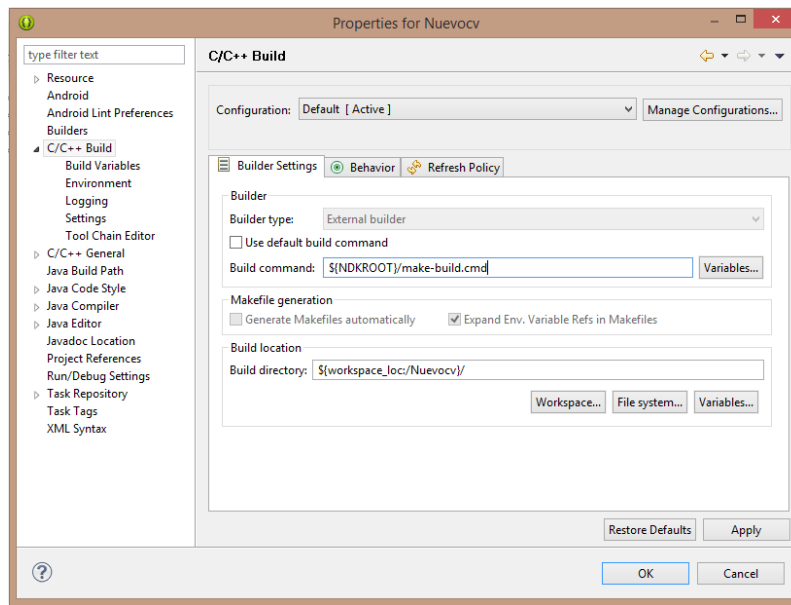


Figura 30. Escritura de comando de ejecución.

11. Luego en la misma ventana seleccionamos la opción C/C++ Build y luego Enviroment y seleccionamos la variable del NDK que habíamos configurado anteriormente (ver Fig. 31).

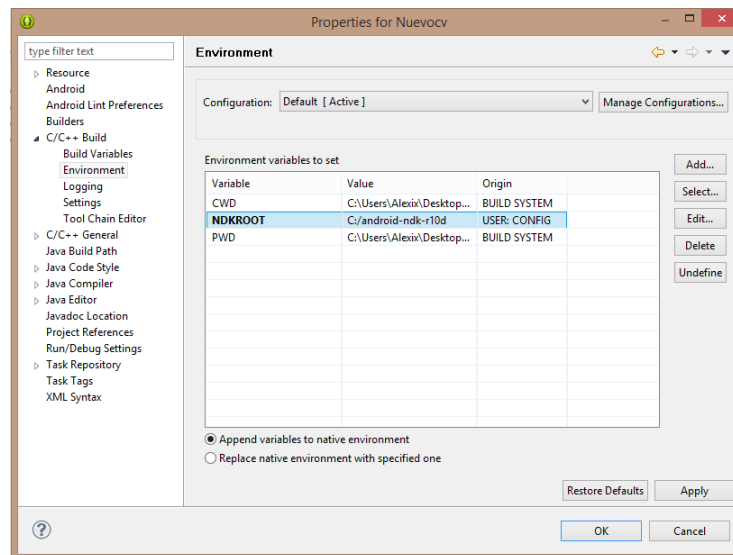


Figura 31. Selección de la variable de entorno NDKROOT.

12. Con esto ya estamos listos para trabajar con OpenCV en Android.

2.9.1 Planificación Inicial

Debemos establecer los pre-requisitos de cada proceso estableciendo las actividades de debe cumplir cada uno de ellos, además se definir una planificación de las fases de la metodología de desarrollo en las cuales se detalla lo que se va hacer a cada iteración.

2.9.1.1 Análisis de Procesos y Pre-requisitos

Para la realización de los procesos es necesario antes cumplir con ciertos pre-requisitos con el fin de implementar la funcionalidad del proceso que describimos a continuación:

P001: Detectar dispositivos bluetooth:

- Detectar los dispositivos Bluetooth.
- Conexión al dispositivo electrónico Arduino una vez que se haya terminado el proceso de detección.
- Verificar si el dispositivo se encuentra configurado o si el dispositivo se encuentra en estado funcional.

P002: Guardar capturas de rostro del usuario:

- Usar cualquiera de las cámaras que posea el dispositivo móvil.
- Poder detectar su rostro al enfocar su cámara.
- Iniciar con el entrenamiento de su patrón facial a través de las cámaras que posea el dispositivo móvil para su posterior identificación.
- Guardar los datos del patrón facial (fotogramas) obtenidos en el entrenamiento dentro del dispositivo Android.

P003: Guardar datos personales de usuario:

- Obtener y almacenar los datos personales del usuario dentro del dispositivo electrónico Arduino.

P004: Guardar datos del dispositivo móvil Android:

- Guardar el nombre y la dirección MAC del dispositivo dentro del dispositivo electrónico Arduino.

P005: Administrar los dispositivos Bluetooth:

- Registro, consulta, actualización y eliminación de dispositivos móviles bluetooth.

P006: Administrar sus datos personales:

- Registro, consulta, actualización y eliminación de datos personales de usuarios.

P007: Administrar rostros de usuarios:

- Registro, consulta, actualización y eliminación de rostros de usuarios.

P008: Usar cámaras:

- Usar cualquiera de las cámaras que posea el dispositivo móvil.
- Poder detectar su rostro al enfocar su cámara.

P009: Verificar identidad de usuario:

- Verificar la identidad con los datos previamente guardados en el entrenamiento dentro del dispositivo electrónico Arduino.
- Desbloquear el dispositivo electrónico Arduino, aún cuando no existan las condiciones de luz necesarias para el reconocimiento facial mediante otro mecanismo seguro.

P010: Permitir y denegar conexiones entrantes:

- Permitir y denegar las conexiones entrantes hacia el dispositivo electrónico Arduino.

P011: Reestablecer configuraciones de fábrica:

- Permitir el reestablecimiento de las configuraciones iniciales de la aplicación móvil y del dispositivo electrónico en caso de que así lo requiera.

2.9.1.2 Planificación de fases

Las fases y las iteraciones que se darán en el proyecto se reflejan en la tabla IX, donde se explica las iteraciones que se realizan para el módulo de configuración inicial, módulo de administración, módulo de entrenamiento y reconocimiento facial y para el módulo de configuración de la aplicación móvil en la etapa de producción y estabilización.

TABLA IX. PLANIFICACIÓN DE FASES.

Fase	Iteración	Descripción
Exploración	Iteración 0	Establecimiento de grupos de interés, requerimientos iniciales, definición del alcance, establecimiento del proyecto.
Inicialización	Iteración 0	Configuración del ambiente de desarrollo, Planificación inicial, planificación de fases.
Producción	Iteración Módulo de Configuración inicial	Implementación Módulo de configuración inicial, Refinamiento y actualización de historias de usuario, Refinamiento de interfaces, Generación y ejecución de pruebas de aceptación.
	Iteración Módulo de administración	Implementación del Módulo de administración, Refinamiento y actualización de historias de usuario, Refinamiento de interfaces, Generación y ejecución de pruebas de aceptación.
	Iteración Módulo de entrenamiento y reconocimiento facial.	Implementación del Módulo de entrenamiento y reconocimiento facial, Refinamiento y actualización de historias de usuario, Refinamiento de interfaces, Generación y ejecución de pruebas de aceptación.
	Iteración Módulo de configuración de la aplicación móvil.	Implementación del Módulo de configuración de la aplicación móvil. , Refinamiento y actualización de historias de usuario, Refinamiento de interfaces, Generación y ejecución de pruebas de aceptación.

Estabilización	Iteración Módulo de Configuración inicial	Refactorización Módulo de Configuración inicial, Refinamiento de interfaces, Generación y ejecución de pruebas de aceptación
	Iteración Módulo de administración	Refactorización del Módulo de administración, Refinamiento de interfaces, Generación y ejecución de pruebas de aceptación.
	Iteración Módulo de entrenamiento y reconocimiento facial.	Refactorización Módulo de entrenamiento y reconocimiento facial, Refinamiento de interfaces, Generación y ejecución de pruebas de aceptación
	Iteración Módulo de configuración de la aplicación móvil.	Refactorización del Módulo de configuración de la aplicación móvil, Refinamiento de interfaces, Generación y ejecución de pruebas de aceptación
Pruebas del Sistema	Iteración Pruebas del Sistema	Evaluación de Pruebas y Análisis de Resultados

2.9.2 Diseño del Sistema

El diseño general de todo el proyecto consta de dos partes:

- Una aplicación móvil android que tendrá la función de detectar al usuario autorizado a través de reconocimiento facial para acceder al vehículo de acuerdo a su rol de usuario.
- Un dispositivo electrónico construido a partir la plataforma de hardware y software libre Arduino con componentes adicionales para abrir y cerrar conexiones que activen el encendido del vehículo.

Los dos componentes se comunicaran de forma inalámbrica a través de Bluetooth donde se propone una arquitectura (Ver Fig. 32).



Figura 32. Diseño del Sistema.

Esta arquitectura se la describe a continuación:

1. El usuario enfoca su rostro en la cámara del dispositivo.
2. El dispositivo móvil android detecta y reconoce el rostro.
3. El dispositivo móvil envía los datos obtenidos por Bluetooth para verificar la identidad del usuario.
4. El dispositivo instalado en el auto recibe los datos.
5. El dispositivo compara los datos enviados por el dispositivo móvil android y determina si puede acceder o no al vehículo.
6. Si se ha verificado los datos y el usuario está autorizado a acceder, el dispositivo electrónico instalado en el sistema de encendido se activa y permite el encendido del vehículo.
7. El usuario puede utilizar normalmente el vehículo.

2.9.2.1 Modelo del Dominio

Se detallan las clases que componen la aplicación móvil que son la base para su desarrollo.
(Ver Fig. 33).

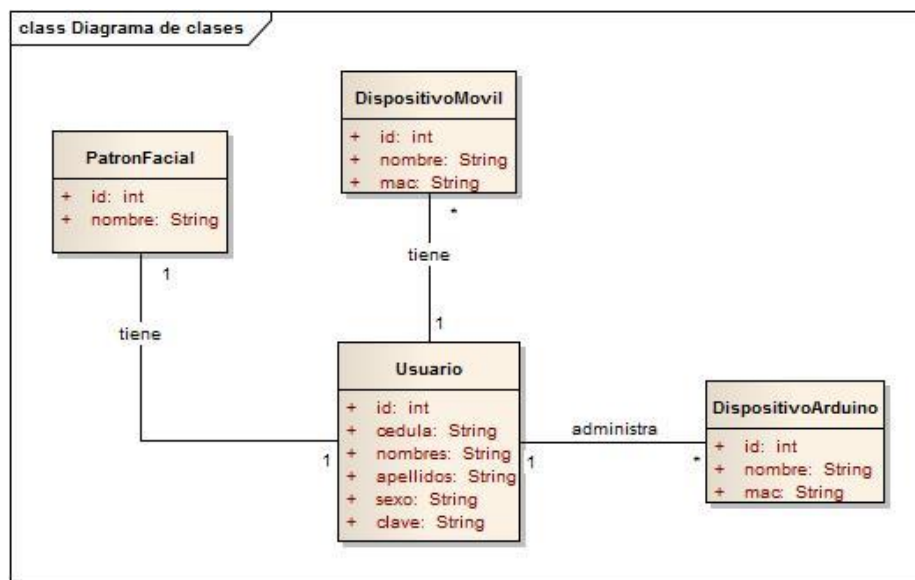


Figura 33. Diagrama de clases

2.9.2.2 Diseño de Base de Datos

A continuación, el esquema diseñado para la base de datos del que sirve como fuente de almacenamiento de la aplicación móvil como se detalla en la Fig. 34:

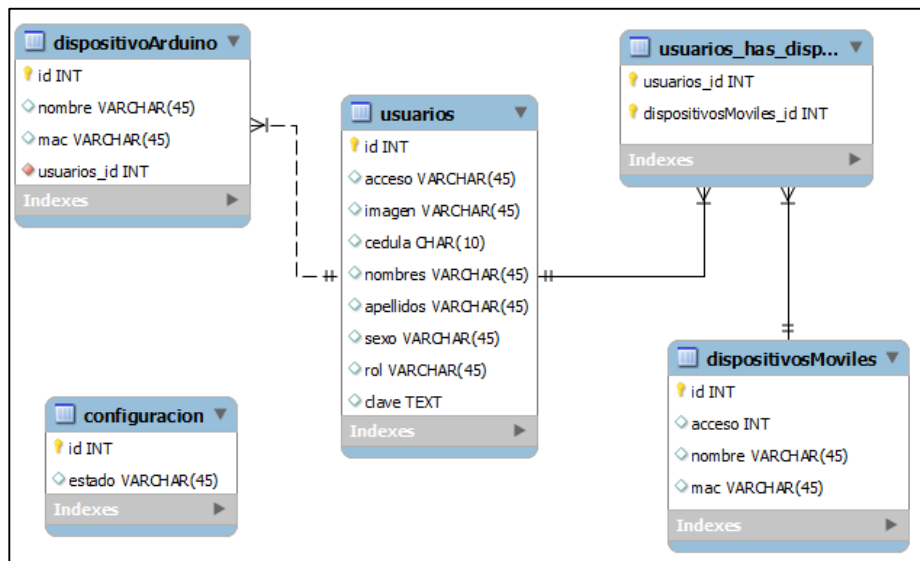


Figura 34. Diseño de base de datos.

2.9.2.3 Descripción de la Interfaz de Usuario

A continuación, se describe el esquema de navegabilidad de la aplicación móvil o storyboard cuyo objetivo es describir la navegabilidad y conexiones entre las principales pantallas de la aplicación desarrollada como podemos observar en la Fig. 35.

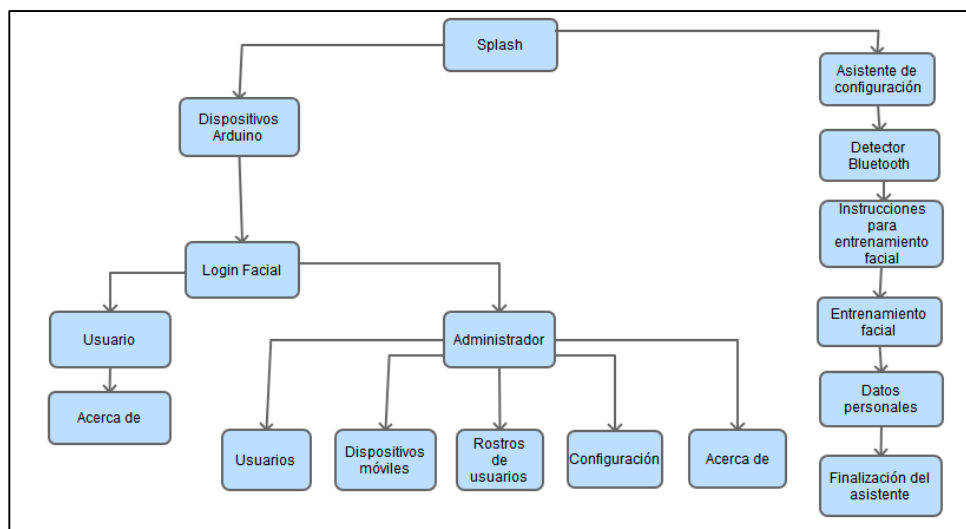


Figura 35. Storyboard de la Aplicación

2.9.2.3.1 Asistente de configuración

La pantalla de Asistente de configuración dará la bienvenida para la configuración inicial al ejecutar la aplicación por primera vez, será presidida por un splash con el logo de la aplicación (Ver Fig. 36):

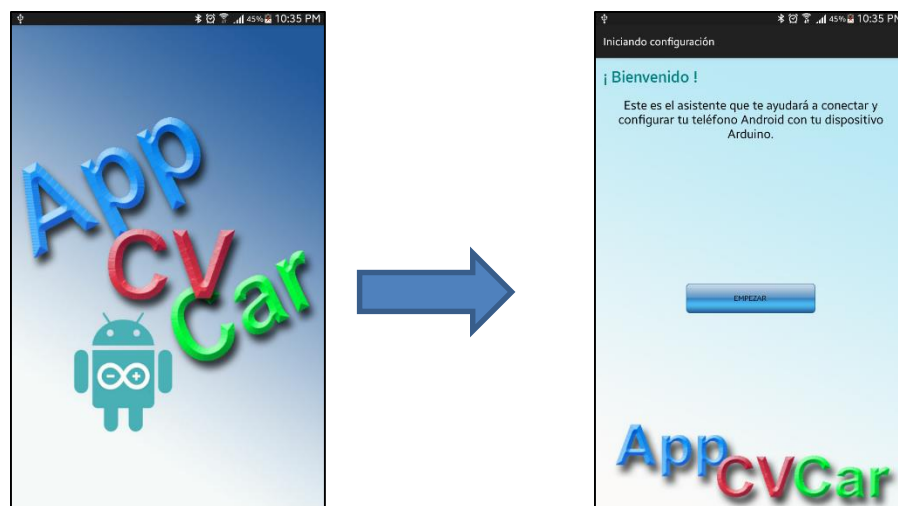


Figura 36. Prototipado de Pantalla: Asistente de configuración.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Asistente de configuración en la tabla X. El storycard muestra una descripción del funcionamiento de la Pantalla Asistente de configuración y de las fechas de su desarrollo.

TABLA X. STORYCARD DE LA PANTALLA ASISTENTE DE INSTALACIÓN.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
01	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Fecha	Estado			Comentario			
06/04/2015	Definido						
18/04/2015	Implementado						
	Verificado						

	Propuesto/ Cancelado/Comparado	
--	---	--

2.9.2.3.2 Detector Bluetooth

La pantalla de Detector Bluetooth mostrará en una lista todos los dispositivos bluetooth que detecte (Ver Fig. 37):

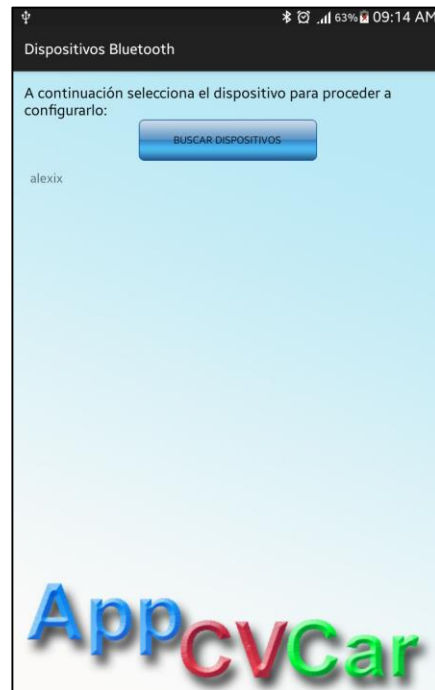


Figura 37. Prototipado de la Pantalla Detector Bluetooth.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Detector Bluetooth en la tabla XI. El storycard muestra una descripción del funcionamiento de la Pantalla Detector Bluetooth y de las fechas de su desarrollo.

TABLA XI. STORYCARD DE LA PANTALLA DETECTOR BLUETOOTH.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
02	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	

Descripción		
<p>Cuando se cargue la pantalla mostrará todos los dispositivos bluetooth detectados.</p> <p>El usuario deberá presionar sobre el dispositivo Arduino que desea seleccionar para conectarse, verificar su estado y poder continuar con el proceso de configuración.</p>		
Excepciones		
<p>Si el dispositivo Arduino no tiene una tarjeta instalada mostrará un mensaje pidiendo que se inserte una tarjeta de memoria.</p> <p>Si el dispositivo Arduino ya se encuentra configurado, mostrará un diálogo con el mensaje de que ya se encuentra configurado, mostrando opciones de registro.</p> <p>Si el usuario presiona el botón atrás se mostrará un diálogo preguntado si desea salir de la aplicación.</p>		
Fecha	Estado	Comentario
06/04/2015	Definido	
18/04/2015	Implementado	
	Verificado	
	Propuesto/ Cancelado/Comparado	

2.9.2.3.3 Instrucciones entrenamiento facial

La pantalla de Instrucciones entrenamiento facial dará indicaciones al usuario para grabar su rostro por primera vez (Ver Fig. 38):

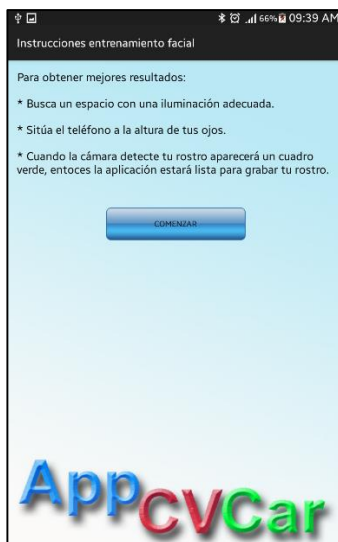


Figura 38. Prototipado de Pantalla: Instrucciones entrenamiento facial.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Instrucciones entrenamiento facial en la tabla XII. El storycard muestra una descripción del funcionamiento de la Pantalla Instrucciones entrenamiento facial y de las fechas de su desarrollo.

TABLA XII. STORYCARD DE PANTALLA INSTRUCCIONES ENTRENAMIENTO FACIAL.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
03	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
Cuando la pantalla se cargue mostrará indicaciones al usuario para un correcto grabado de rostro. El usuario después de leer las instrucciones deberá presionar en el botón Empezar para continuar.							
Excepciones							
Si el usuario presiona el botón atrás se mostrará un diálogo preguntado si desea salir de la aplicación.							
Fecha	Estado			Comentario			
06/04/2015	Definido						
19/04/2015	Implementado						
	Verificado						
	Propuesto/ Cancelado/Comparado						

2.9.2.3.4 Entrenamiento facial

La pantalla de Entrenamiento facial será la encargada de tomar los fotogramas del usuario para después usarlos para el reconocimiento del rostro del usuario (Ver Fig. 39):

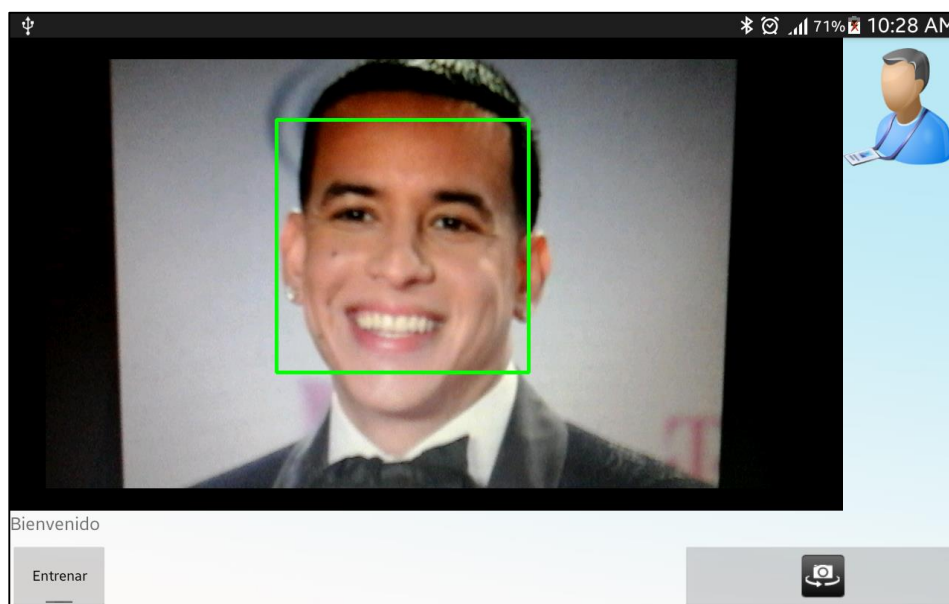


Figura 39. Prototipado de Pantalla: Asistente de configuración.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Entrenamiento facial en la tabla XIII. El storycard muestra una descripción del funcionamiento de la Pantalla Entrenamiento facial y de las fechas de su desarrollo.

TABLA XIII. STORYCARD DE PANTALLA ENTRENAMIENTO FACIAL.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
04	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
El usuario deberá colocar la cámara del dispositivo a la altura de los ojos de manera que esta detecte su rostro y muestre un cuadro verde sobre el mismo.							
Cuando el cuadro verde haya aparecido deberá presionar en el botón entrenar, en ese momento se activará el botón Grabar.							
El usuario deberá presionar en el botón Grabar con lo cual la aplicación móvil comenzará a hacer capturas del rostro.							
Para terminar con el entrenamiento hay que presionar en el botón Detener entrenamiento.							

Se recomienda hacer diferentes gestos con la cara para un mejor resultado en el reconocimiento facial.		
Excepciones		
Si el usuario presiona el botón atrás se mostrará un diálogo preguntado si desea salir de la aplicación.		
Fecha	Estado	Comentario
07/04/2015	Definido	
19/04/2015	Implementado	
	Verificado	
	Propuesto/ Cancelado/Comparado	

2.9.2.3.5 Datos personales

La pantalla de Datos personales será la encargada de guardar los datos del usuario para la posterior identificación (Ver Fig. 40):

La imagen muestra un prototipo de la interfaz de usuario para la pantalla 'Datos personales' de la aplicación 'AppCVCar'. La pantalla tiene un encabezado con el título 'Datos personales'. Los campos de entrada incluyen: 'Cédula:', 'Nombres:', 'Apellidos:', 'Sexo:' (con opciones 'Hombre' seleccionada y 'Mujer'), 'Contraseña:', y 'Confirmar:'. Debajo de estos campos hay un botón azul con el texto 'GUARDAR'. En la parte inferior de la pantalla, se encuentra el logo 'AppCVCar' en colores azul, rojo y verde.

Figura 40. Prototipado de Pantalla: Datos personales.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Datos personales en la tabla XIV. El storycard muestra una descripción del funcionamiento de la Pantalla Datos personales y de las fechas de su desarrollo.

TABLA XIV. STORYCARD DE LA PANTALLA DATOS PERSONALES.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
05	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
<p>Cuando se cargue la pantalla aparecerá un diálogo indicando que todos los campos son obligatorios, el usuario tendrá que ingresar todos los datos que se solicita.</p> <p>La clave no puede tener menos de 8 caracteres.</p> <p>El usuario deberá presionar en el botón Guardar para continuar.</p>							
Excepciones							
<p>Si algún campo está vacío se presentará un mensaje indicando que campo no se ha ingresado.</p> <p>Se mostrará un dialogo con el mensaje Datos no guardados en caso de que no se hayan podido guardar los datos.</p> <p>Si el usuario presiona el botón atrás se mostrará un dialogo preguntado si desea salir de la aplicación.</p>							
Fecha	Estado			Comentario			
07/04/2015	Definido						
28/04/2015	Implementado						
	Verificado						
	Propuesto/ Cancelado/Comparado						

2.9.2.3.6 Finalizar configuración

La pantalla de Finalizar configuración mostrará al usuario un mensaje de que la configuración ha finalizado con éxito. (Ver Fig. 41):

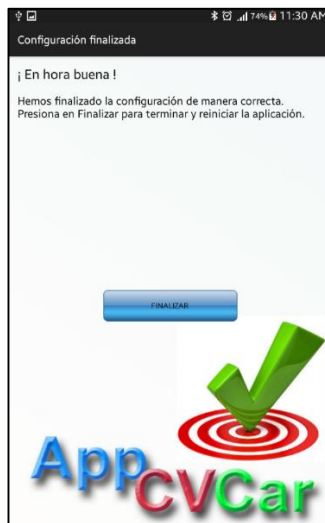


Figura 41. Prototipado de Pantalla: Finalizar configuración.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Finalizar configuración en la tabla XV. El storycard muestra una descripción del funcionamiento de la Pantalla Finalizar configuración y de las fechas de su desarrollo.

TABLA XV. STORYCARD DE LA PANTALLA FINALIZAR CONFIGURACIÓN.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
06	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
<p>Cuando se cargue la pantalla mostrará al usuario un mensaje indicando que la configuración ha finalizado con éxito.</p> <p>El usuario deberá presionar en el botón Finalizar para terminar con la configuración y reiniciar la aplicación.</p>							
Excepciones							
Si el usuario presiona el botón atrás se mostrará un dialogo preguntado si desea salir de la aplicación.							
Fecha	Estado			Comentario			
07/04/2015	Definido						

29/04/2015	Implementado	
	Verificado	
	Propuesto/ Cancelado/Comparado	

2.9.2.3.7 Dispositivos Arduino

La pantalla de Dispositivos Arduino es la que nos administran los dispositivos Arduino que registremos en la aplicación, es decir, permitirá desbloquear más de un dispositivo Arduino. (Ver Fig. 42):

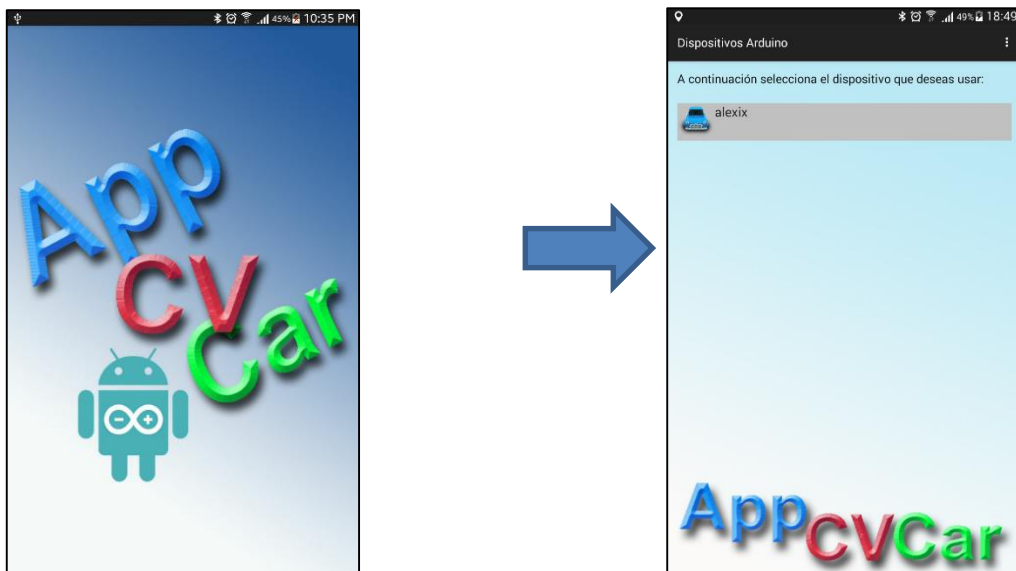


Figura 42. Prototipado de Pantalla: Dispositivos Arduino.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Dispositivos Arduino en la tabla XVI. El storycard muestra una descripción del funcionamiento de la Pantalla Dispositivos Arduino y de las fechas de su desarrollo.

TABLA XVI. STORYCARD DE PANTALLA DISPOSITIVO ARDUINO.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
07	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
<p>Cuando el usuario ingrese a la aplicación se cargará un splash con el logo de la aplicación, luego se cargará la pantalla Dispositivos Arduino donde el usuario podrá elegir que dispositivo desea desbloquear</p> <p>Deberá presionar sobre el dispositivo y acto seguido seleccionar la opción desbloquear para continuar.</p>							
Excepciones							
Si solo hay un dispositivo registrado no se podrá eliminarlo.							
Fecha	Estado			Comentario			
08/04/2015	Definido						
01/05/2015	Implementado						
	Verificado						
	Propuesto/ Cancelado/Comparado						

2.9.2.3.8 Logueo facial

La pantalla de Logueo facial es la que nos permitirá acceder a la aplicación móvil y por tanto también al dispositivo Arduino que se encuentra instalado el automóvil. (Ver Fig. 43):

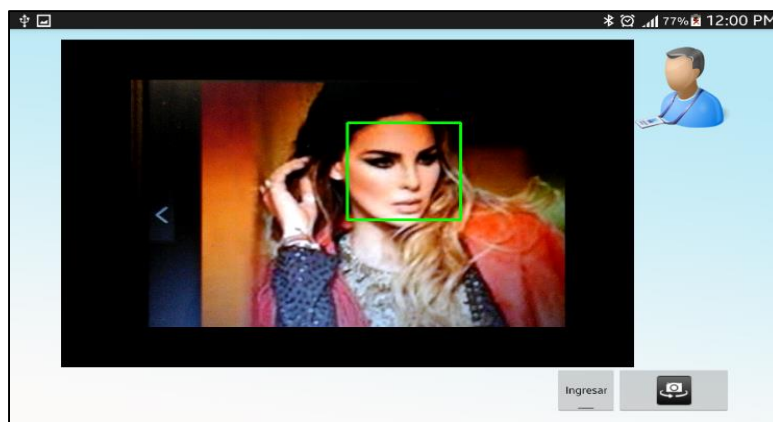


Figura 43. Prototipado de Pantalla: Logueo facial.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Logueo facial en la tabla XVII. El storycard muestra una descripción del funcionamiento de la Pantalla Logueo facial y de las fechas de su desarrollo.

TABLA XVII. STORYCARD DE PANTALLA LOGUEO FACIAL.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
08	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
Cuando el usuario haya elegido el dispositivo Arduino, se cargará la pantalla Logueo facial donde tendrá que enfocar su rostro para que la aplicación móvil lo reconozca y le permita acceder. Deberá presionar el botón Ingresar para comenzar con el reconocimiento facial.							
Excepciones							
Si la aplicación móvil no lo reconoce mostrará un diálogo informando que su acceso no está autorizado y activará un botón adicional que le permitirá ingresar a través de la contraseña que ingreso anteriormente.							
Fecha	Estado			Comentario			
08/04/2015	Definido						
01/05/2015	Implementado						
	Verificado						
	Propuesto/ Cancelado/Comparado						

2.9.2.3.9 Logueo clave

La pantalla de Logueo clave permitirá al usuario ingresar a través de cédula y clave en caso que la aplicación no lo reconozca o no hayan las condiciones de iluminación suficientes (Ver Fig. 44):

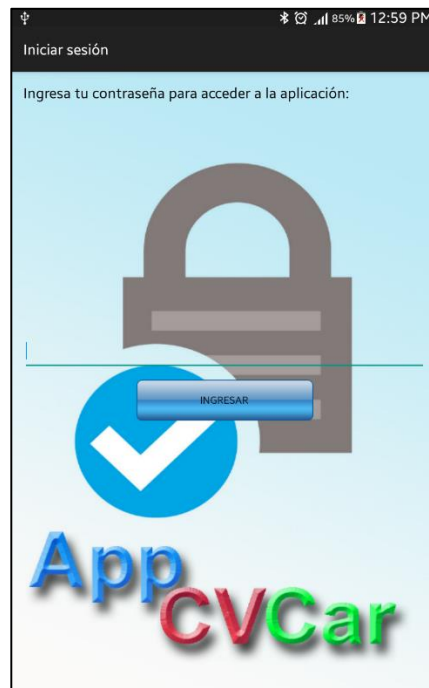


Figura 44. Prototipado de Pantalla: Logueo clave.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Logueo clave en la tabla XVIII. El storycard muestra una descripción del funcionamiento de la Pantalla Asistente de configuración y de las fechas de su desarrollo.

TABLA XVIII. STORYCARD DE PANTALLA LOGUEO CLAVE

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
09	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
<p>Cuando el usuario no pueda acceder mediante su rostro tendrá la opción de acceder mediante su cédula y contraseña que ingreso al momento de la configuración.</p> <p>Al ingresar la cédula y contraseña tendrá que presionar en el botón Ingresar con lo cual la aplicación validará los datos y permitirá el acceso.</p>							
Excepciones							
Si el usuario no existe presentará un diálogo con el mensaje que el usuario no existe.							

Si el usuario no tiene permitido el acceso presentará un diálogo con el mensaje No estás autorizado para ingresar al sistema.		
Fecha	Estado	Comentario
08/04/2015	Definido	
15/05/2015	Implementado	
	Verificado	
	Propuesto/ Cancelado/Comparado	

2.9.2.3.10 Administrador

La pantalla de Administrador será la que contiene el acceso a todas las funciones de la aplicación y el dispositivo Arduino y solo podrá ser accedida por usuarios con permisos de Administrador (Ver Fig. 45):

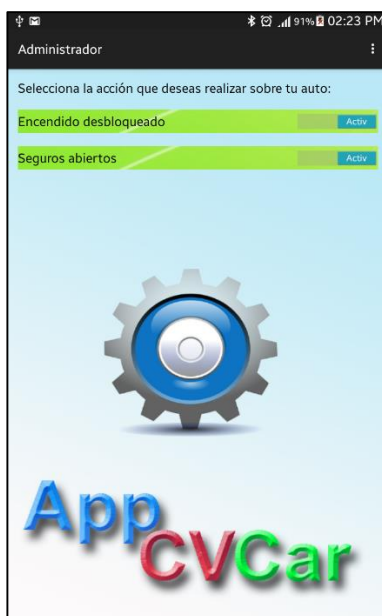


Figura 45. Prototipado de Pantalla: Administrador.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Administrador en la tabla XIV. El storycard muestra una descripción del funcionamiento de la Pantalla Administrador y de las fechas de su desarrollo.

TABLA XIX. STORYCARD DE PANTALLA ADMINISTRADOR.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
10	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
<p>Cuando el usuario se ha logueado como administrador se cargan todas las funciones de la aplicación, activando el dispositivo Arduino el cual está bloqueando el automovil.</p> <p>El administrador tendrá la opción de activar o desactivar el dispositivo Arduino a voluntad siempre y cuando mantenga activa la sesión en la aplicación.</p>							
Excepciones							
Si el usuario cierra sesión en esta pantalla se desactivará el dispositivo Arduino con lo cual bloquearía el automovil donde se encuentre instalado.							
Fecha	Estado			Comentario			
09/04/2015	Definido						
16/05/2015	Implementado						
	Verificado						
	Propuesto/ Cancelado/Comparado						

2.9.2.3.11 Administrar usuarios

La pantalla de Administrar usuarios brinda las opciones de consulta, creación, actualización y eliminación de usuarios, así como también la opción de denegar el acceso al usuario que sea necesario (Ver Fig. 46):

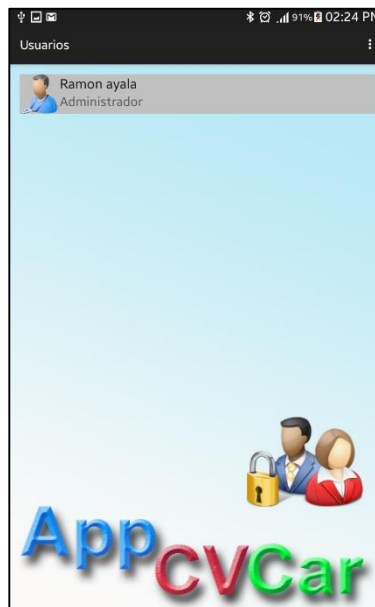


Figura 46. Prototipado de Pantalla: Asistente de configuración.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Administrar usuarios en la tabla XX. El storycard muestra una descripción del funcionamiento de la Pantalla Administrar usuarios y de las fechas de su desarrollo.

TABLA XX. STORYCARD DE PANTALLA ADMINISTRAR USUARIOS.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
11	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
<p>Cuando el usuario selecciona la opción Usuarios de aplicación del menú de la pantalla Administrador se carga la pantalla de Administrar usuarios con todos los usuarios que maneja la aplicación.</p> <p>Para crear un nuevo usuario se selecciona la opción Nuevo usuario del menú del ActionBar y se sigue las instrucciones donde hay que agregar los datos personales del nuevo usuario y su rostro.</p> <p>Para actualizar los datos de un usuario se selecciona dicho usuario de la lista y esto nos direccionará automáticamente a la pantalla para actualizar los datos.</p>							

Para eliminar un usuario en la misma pantalla de actualización se selecciona la opción de eliminar en el menú del ActionBar.

Los datos personales del usuario estarán almacenados dentro del dispositivo Arduino.

Excepciones

Si hay un único usuario de la aplicación no se podrá eliminar ni bloquear su acceso.

Para denegar el acceso a un usuario siempre se tendrá que cambiar la clave de dicho usuario por motivos de seguridad.

Fecha	Estado	Comentario
17/04/2015	Definido	
	Implementado	
	Verificado	
	Propuesto/ Cancelado/Comparado	

2.9.2.3.12 Dispositivos móviles

La pantalla de Dispositivos móviles tiene las funciones de eliminar y bloquear el acceso del dispositivo móvil seleccionado, ya que este se registra automáticamente en el asistente de instalación con el fin de disminuir el trabajo para el usuario (Ver Fig. 47):



Figura 47. Prototipado de Pantalla: Dispositivos móviles

A continuación, se describe el StoryCard del Prototipado de Pantalla: Dispositivos móviles en la tabla XXI. El storycard muestra una descripción del funcionamiento de la Pantalla Dispositivos móviles y de las fechas de su desarrollo.

TABLA XXI. STORYCARD DE PANTALLA DISPOSITIVOS MÓVILES.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
12	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
<p>Cuando el usuario selecciona la opción Dispositivos móviles del menú de la pantalla Administrador se carga la pantalla de Dispositivos móviles con todos los dispositivos móviles que maneja la aplicación.</p> <p>Para eliminar o bloquear un dispositivo móvil se selecciona dicho dispositivo en la lista que direcciona hacia otra pantalla donde se pueden realizar estas acciones.</p> <p>Los datos personales de los dispositivos móviles estarán almacenados dentro del dispositivo Arduino.</p>							
Excepciones							
Si hay un único dispositivo móvil no se podrá eliminar ni bloquear su acceso.							
Fecha	Estado			Comentario			
10/04/2015	Definido						
22/05/2015	Implementado						
	Verificado						
	Propuesto/ Cancelado/Comparado						

2.9.2.3.13 Rostros de Usuarios

La pantalla de Rostros de usuarios muestra todos los rostros de usuario que maneja la aplicación en la cual las acciones que se pueden realizar es ver, actualizar y eliminar los rostros almacenados (Ver Fig. 48):

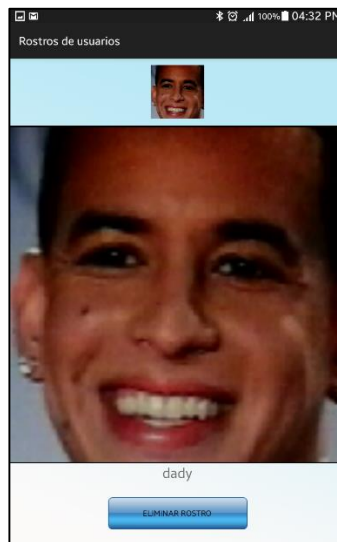


Figura 48. Prototipado de Pantalla: Rostros de usuarios.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Rostros de usuarios en la tabla XXII. El storycard muestra una descripción del funcionamiento de la Pantalla Rostros de usuarios y de las fechas de su desarrollo.

TABLA XXII. STORYCARD DE PANTALLA ROSTROS DE USUARIO.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
13	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
<p>Cuando el usuario selecciona la opción Rostros de usuario del menú de la pantalla Administrador se carga la pantalla Rostros de usuario con todos los rostros de usuario que maneja la aplicación.</p> <p>Cuando el usuario ingrese los rostros se cargarán en la parte superior de la pantalla, para ver cada uno de ellos solo hay que seleccionar el rostro que se desee.</p> <p>Para eliminar un rostro basta con seleccionarlo y presionar en el botón Eliminar el cual presentará un diálogo preguntando Esta seguro de que desea eliminar permanentemente este usuario.</p> <p>Para actualizar un rostro basta con seleccionarlo y presionar en el botón Actualizar el cual presentará una nueva pantalla para actualizar el rostro</p> <p>Si se elimina el rostro también se eliminará los datos personales asociados a ese rostro.</p>							

Excepciones		
Si hay un único usuario de la aplicación no se podrá eliminar.		
Fecha	Estado	Comentario
10/04/2015	Definido	
29/05/2015	Implementado	
	Verificado	
	Propuesto/ Cancelado/Comparado	

2.9.2.3.14 Ajustes de aplicación

La pantalla de Ajustes de aplicación permitirá realizar ajustes como denegar las conexiones hacia el dispositivo Arduino y en caso de ser necesario reestablecer los ajustes de fábrica de la aplicación y el dispositivo Arduino (Ver Fig. 49):



Figura 49. Prototipado de Pantalla: Asistente de configuración.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Ajustes de aplicación en la tabla XXIII. El storycard muestra una descripción del funcionamiento de la Pantalla Asistente de configuración y de las fechas de su desarrollo.

TABLA XXIII. STORYCARD DE PANTALLA AJUSTES DE APLICACIÓN

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
14	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
<p>Cuando el usuario selecciona la opción Ajustes del menú de la pantalla Administrador se carga la pantalla Ajustes de aplicación.</p> <p>Las conexiones entrantes están desactivadas por defecto, para activarlas basta con deslizar el switch.</p> <p>Para reestablecer los ajustes de fábrica presionamos el botón Reestablecer todo, el cual nos mostrará un diálogo preguntando Estas seguro que deseas eliminar todos los datos, presionando en confirmar todos los datos se eliminarán.</p>							
Excepciones							
Las conexiones entrantes se desactivaran automáticamente cuando se agregue un dispositivo móvil.							
Fecha	Estado			Comentario			
13/04/2015	Definido						
06/06/2015	Implementado						
	Verificado						
	Propuesto/ Cancelado/Comparado						

2.9.2.3.15 Usuario

La pantalla de Usuario no contiene funciones de administración, diseñada solo para el desbloqueo del automóvil, pensada en usuarios normales (Ver Fig. 50):

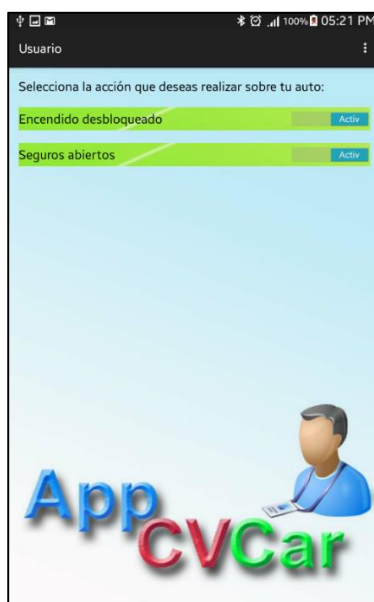


Figura 50. Prototipado de Pantalla: Asistente de configuración.

A continuación, se describe el StoryCard del Prototipado de Pantalla: Usuario en la tabla XXIV. El storycard muestra una descripción del funcionamiento de la Pantalla Usuario y de las fechas de su desarrollo.

TABLA XXIV. STORYCARD DE PANTALLA USUARIO.

Numero/Id	Tipo	Dificultad		Esfuerzo		Prioridad	Nota
		Antes	Después	Estimado	Gastado		
15	Nuevo	Fácil	Fácil			Baja	
	Fijo	Moderado	Moderado			Media	
	Mejora	Duro	Duro			Alta	
Descripción							
Cuando el usuario se haya logueado se cargará la pantalla de Usuario y se activará el dispositivo Arduino desbloqueando el automovil donde se encuentre instalado. El usuario podrá bloquearlo y desbloquearlo a voluntad.							
Excepciones							
Si el usuario cierra la sesión el dispositivo Arduino se desactivará automáticamente.							
Fecha	Estado			Comentario			
13/04/2015	Definido						
10/05/2015	Implementado						

	Verificado	
	Propuesto/ Cancelado/Comparado	

2.9.3 Diseño del Dispositivo Arduino

En esta sección se describirá todos los materiales, diseño electrónico, esquemas de conexión y capturas de todo el proceso de construcción del dispositivo arduino.

2.9.3.1 Materiales Hardware

En la tabla XXV se describen todos los materiales utilizados en el prototipo.

TABLA XXV. MATERIALES HARDWARE

Cantidad	Descripción
1	Arduino Mega 2560
1	Lector de tarjetas SD
1	Módulo bluetooth HC-06
2	Módulo Relay Arduino
3	Resistencias de 200 kΩ
3	Led colores Alta luminosidad
1	Batería 9V
12	Pares de cable macho-hembra

2.9.3.2 Esquemas de conexión

En este apartado se mostrarán los esquemas utilizados en la conexión de los diferentes componentes del prototipo detallando todo el proceso de construcción.

2.9.3.2.1 Conexión Arduino Bluetooth.

En la Fig. 51 podemos apreciar de forma gráfica las conexiones realizadas entre el Arduino y el módulo bluetooth.

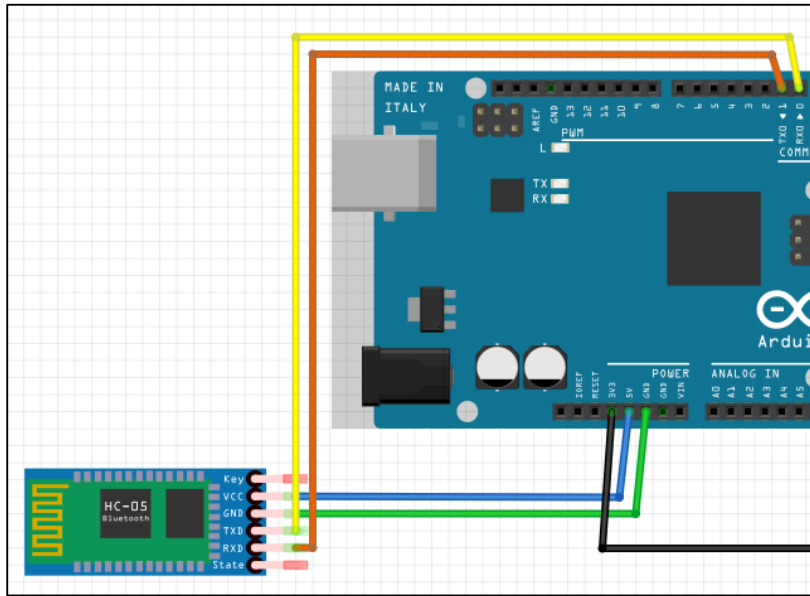


Figura 51. Esquema Conexión Arduino bluetooth.

La tabla XXVI contiene la especificación de los pines utilizados en la conexión del Arduino y el módulo bluetooth.

TABLA XXVI. CONEXIÓN DE PINES ARDUINO BLUETOOTH.

Pin Arduino	Pin bluetooth
5V	VCC
GND	GND
TX	RX
RX	TX

2.9.3.2.2 Conexión Arduino Lector SD.

En la Fig. 52 podemos apreciar de forma gráfica las conexiones realizadas entre el Arduino y el módulo lector de SD.

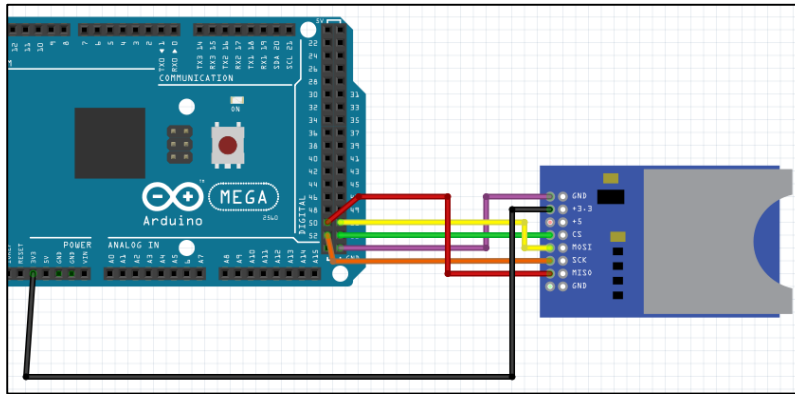


Figura 52. Esquema conexión Arduino Lector SD.

La tabla XXVII contiene la especificación de los pines utilizados en la conexión del Arduino y el módulo lector de SD.

TABLA XXVII. CONEXIÓN DE PINES ARDUINO LECTOR SD

Pin Arduino	Pin Lector SD
Pin 50	MISO
Pin 51	MOSI
Pin 52	SCK
Pin 53	CS
Pin 3.3V	Pin 3.3V
GND	GND

2.9.3.2.3 Conexión Arduino Módulos Relay

En la Fig. 53 podemos apreciar de forma gráfica las conexiones realizadas entre el Arduino y los módulos Relay.

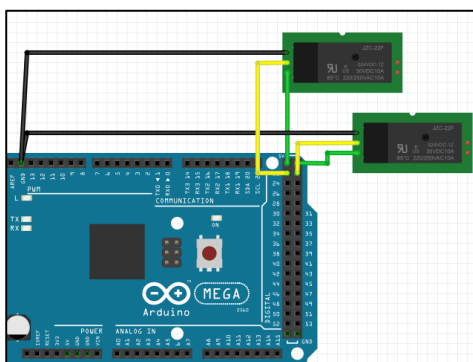


Figura 53. Esquema conexión Arduino Módulos Relay.

La tabla XXVIII y tabla XXIX la especificación de los pines utilizados en la conexión del Arduino y los módulos Relay.

TABLA XXVIII. CONEXIÓN DE PINES ARDUINO MODULO RELAY 1

Pin Arduino	Pin Lector SD
5V	DC+
GND	DC-
Pin 22	IN

TABLA XXIX. CONEXIÓN DE PINES ARDUINO MODULO RELAY 2

Pin Arduino	Pin Lector SD
5V	DC+
GND	DC-
Pin 23	IN

2.9.3.2.4 Conexión Arduino Leds estado

En la Fig. 54 podemos apreciar de forma gráfica las conexiones realizadas entre el Arduino y los led que indicarán el estado del dispositivo en funcionamiento.

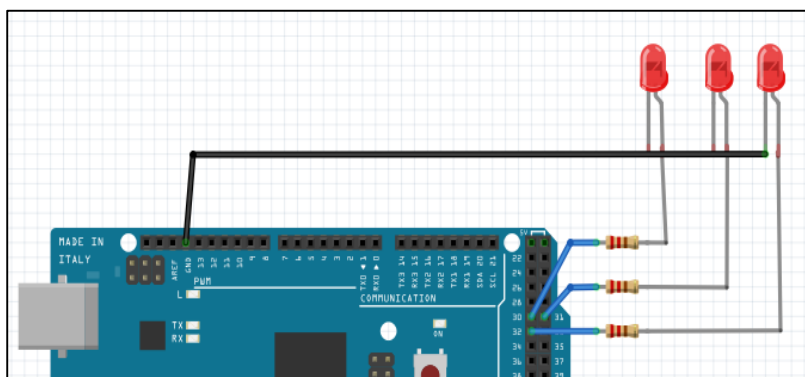


Figura 54. Esquema conexión Arduino Leds estado

La tabla XXX, tabla XXXI y tabla XXXII contienen la especificación de los pines utilizados en la conexión del Arduino y los led.

TABLA XXX. CONEXIÓN DE PINES ARDUINO CON LED ACTIVIDAD

Pin Arduino	Pin Lector SD
5V	DC+
GND	DC-
Pin 23	IN

TABLA XXXI. CONEXIÓN DE PINES ARDUINO CON LED BLOQUEO

Pin Arduino	Pin Lector SD
5V	DC+
GND	DC-
Pin 23	IN

Tabla XXXII. CONEXIÓN DE PINES ARDUINO CON LED SEGUROS

Pin Arduino	Pin Lector SD
5V	DC+
GND	DC-
Pin 24	IN

2.9.3.3 Esquema final de conexiones.

En la Fig. 55 podemos apreciar de forma gráfica todas las conexiones realizadas entre el Arduino y todos los componentes mostrados anteriormente.

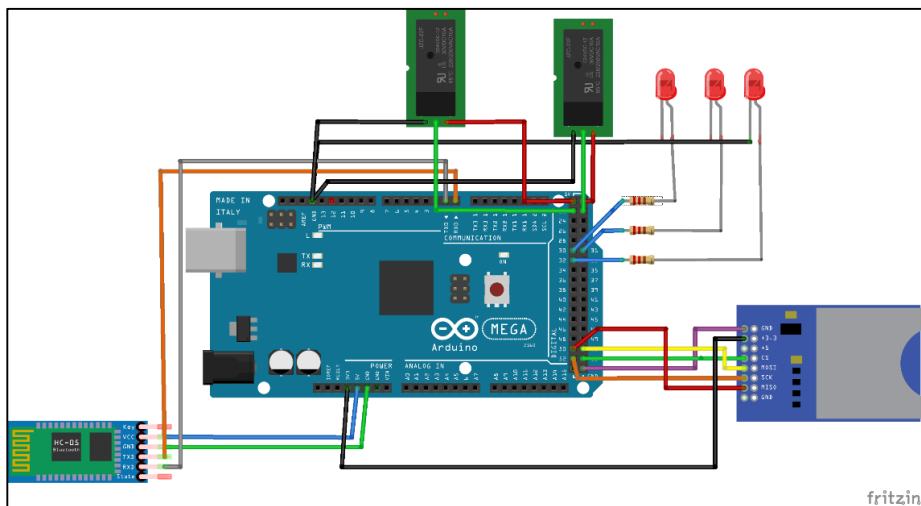


Figura 55. Esquema final de conexiones.

2.9.3.4 Vista real de prototipo construido.

En la Fig. 56 y Fig. 57 podemos apreciar una vista real del prototipo realizado previo a la puesta en marcha de su funcionamiento.



Figura 56. Vista superior del prototipo.



Figura 57. Vista Lateral del prototipo.

3. CODIFICACIÓN

En esta fase se ha realizado el desarrollo de las funcionalidades que se plantearon en el diseño siendo este el caso de Android el cual utiliza java; además de que se plantean estándares de codificación y la estructuración de la aplicación.

3.1 Producción y estabilización

3.1.1 Estándares de Codificación

- **Paquetes:** Los paquetes están escritos en minúsculas.
- **Clases:** Las clases están escritas con la primera letra en mayúscula, si se compone de dos o más palabras la primera letra de cada una de ellas inicia en mayúscula.
- **Variables:** Las variables se han nombrado de acuerdo a su valor, están escritas con la primera letra en minúscula, si se compone de dos o más palabras la primera letra de ellas inicia en mayúscula.
- **Métodos:** Los métodos llevan un nombre relacionado al proceso que ejecutan, están escritos con la primera letra en minúscula, si se compone de dos o más palabras la primera letra de ellos inicia en mayúscula.
- **Layouts:** Los layouts tienen todo su nombre en minúsculas, si poseen más de dos palabras tendrán un guión bajo para separar cada una de ellas.

3.1.2 Estructura de Directorios

A continuación, en la Fig. 58, se describe la estructura de la aplicación y una descripción de los directorios que contiene la aplicación móvil.



Figura 58. Estructura de la aplicación móvil.

3.1.3 Codificación

El desarrollo del código se realizó basándose en los storycards realizados en la fase anterior, pero debido a la extensión del código se colocará el código más importante.

Splash

Para la construcción del splash se creó una actividad llamada splash y se implementó métodos para verificar el estado de configuración de la aplicación, la cual determinará que actividad debe llamarse de acuerdo a la información que obtenga de la base de datos (ver Fig. 59).

```

private BluetoothAdapter bluetoothAdapter;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    requestWindowFeature(Window.FEATURE_NO_TITLE);
    setContentView(R.layout.activity_splash);

    AppCVBD bd=new AppCVBD(getApplicationContext());
    bd.insertarEstado(1, "");
    bluetoothAdapter=BluetoothAdapter.getDefaultAdapter();

    Thread timer = new Thread() {
        public void run() {
            try {
                sleep(2000);
            } catch (InterruptedException e) {
                e.printStackTrace();
            } finally {

                AppCVBD bd=new AppCVBD(getApplicationContext());
                String dispo="";
                dispo=bd.recuperarEstado(1);

                if (dispo.isEmpty()) {
                    Intent asistente=new
Intent(getApplicationContext(),Asistente.class);
                    startActivity(asistente);
                    Splash.this.finish();
                } else {

                    Intent logueo=new
Intent(getApplicationContext(),LogueoFacial.class);
                    startActivity(logueo);
                    Splash.this.finish();

                    if ( !bluetoothAdapter.isEnabled() ) {
                        Intent turnOnIntent = new Intent(
BluetoothAdapter.ACTION_REQUEST_ENABLE );
                        startActivityForResult( turnOnIntent, 1
);
                    }
                }
            }
        }
    };

    timer.start();
}

```

Figura 59. Código del splash

Código de envío y recepción de datos Bluetooth

Para el envío y recepción de datos por bluetooth se ha utilizado el siguiente código el cual trabaja en modo half-duplex (ver Fig. 60).

```
public String enviarDatos(String datos,BluetoothDevice device) {

    // TODO: Enviando información del móvil hacia el Arduino.
    OutputStream mmOutputStream=null;
    InputStream mmInputStream=null;
    BufferedReader lectorReader=null;
    String respuestaArduino="";

    String mmUUID = "00001101-0000-1000-8000-00805F9B34FB";
    try {
        clientSocket=device.createInsecureRfcommSocketToServiceRecord(U
UID.fromString(mmUUID));
        clientSocket.connect();
    } catch (IOException e1) {
        respuestaArduino="no_conectado";
    }

    try {
        if (clientSocket.isConnected()) {
            mmOutputStream=clientSocket.getOutputStream();
            mmOutputStream.write(new String(datos).getBytes());
            mmInputStream=clientSocket.getInputStream();

            try {
                lectorReader=new BufferedReader(new
InputStreamReader(mmInputStream));
                respuestaArduino=lectorReader.readLine();
            } catch (Exception e) {
                respuestaArduino="no_conectado";
            }
            clientSocket.close();
            return respuestaArduino;
        }
    } catch (IOException e) {
        respuestaArduino="no_conectado";
    }
    return respuestaArduino;
}
```

Figura 60. Código de envío y recepción de datos por Bluetooth.

Código para el cifrado de información.

Para el envío de información a través de bluetooth se ha considerado ocultar la información que se transmite usando el algoritmo MD5 como se puede ver en la Fig. 61.

```
private static final char[] CONSTS_HEX = { '0', '1', '2', '3', '4',  
      '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f' };  
  
public String cifrarMD5(String stringEncriptar){  
    try {  
        MessageDigest msgd = MessageDigest.getInstance("MD5");  
        byte[] bytes = msgd.digest(stringEncriptar.getBytes());  
        StringBuilder strbCadenaMD5 = new StringBuilder(2 *  
bytes.length);  
        for (int i = 0; i < bytes.length; i++) {  
            int bajo = (int) (bytes[i] & 0x0f);  
            int alto = (int) ((bytes[i] & 0xf0) >> 4);  
            strbCadenaMD5.append(CONSTS_HEX[alto]);  
            strbCadenaMD5.append(CONSTS_HEX[bajo]);  
        }  
        return strbCadenaMD5.toString();  
    } catch (NoSuchAlgorithmException e) {  
        return null;  
    }  
}
```

Figura 61. Algoritmo MD5.

Código de detección y reconocimiento de rostros.

Para la detección y reconocimiento de rostros se ha utilizado la herramienta OpenCV Android la cual facilita esta tarea, en este apartado se muestra el código utilizado.

Código de detección de rostros

En la detección de rostros al enfocar la cámara sobre un rostro se colocará un cuadro verde sobre el mismo lo que indica que se ha detectado un rostro, el código se muestra en la Fig. 62.

```
private BaseLoaderCallback mLoaderCallback = new
BaseLoaderCallback(this) {
    @Override
    public void onManagerConnected(int status) {
        switch (status) {
            case LoaderCallbackInterface.SUCCESS:{
                Log.i(TAG, "OpenCV loaded successfully");
                fr=new PersonRecognizer(mPath);
                fr.load();

                try {
                    InputStream is =
getResources().openRawResource(R.raw.lbpcascade_frontalface);
                    File cascadeDir = getDir("cascade",
Context.MODE_PRIVATE);
                    mCascadeFile = new File(cascadeDir,
"lbpcascade.xml");
                    FileOutputStream os = new
FileOutputStream(mCascadeFile);

                    byte[] buffer = new byte[4096];
                    int bytesRead;
                    while ((bytesRead = is.read(buffer)) != -1) {
                        os.write(buffer, 0, bytesRead);
                    }
                    is.close();
                    os.close();
                    mJavaDetector = new
CascadeClassifier(mCascadeFile.getAbsolutePath());
```



```

        if (mJavaDetector.empty()) {
            Log.e(TAG, "Failed to load cascade
classifier");
            mJavaDetector = null;
        } else
            Log.i(TAG, "Loaded cascade classifier from
" + mCascadeFile.getAbsolutePath());
            cascadeDir.delete();
        } catch (IOException e) {
            e.printStackTrace();
            Log.e(TAG, "Failed to load cascade. Exception
thrown: " + e);
        }
        mOpenCvCameraView.enableView();

    } break;
    default:{
        super.onManagerConnected(status);
    } break;
}
};

```

Figura 62. Código de detección de rostros.

Código de entrenamiento de rostros

Para el entrenamiento de rostros lo que se hace es tomar fotos del rostro de usuario en este caso se ha considerado 50 fotos, las cuales se almacenan en un directorio de la aplicación móvil, para posteriormente utilizarlas para el reconocimiento, el código se muestra en la Fig. 63.

```

public boolean train() {

    File root = new File(mPath);

    FilenameFilter pngFilter = new FilenameFilter() {
        public boolean accept(File dir, String name) {
            return name.toLowerCase().endsWith(".jpg");
        }
    };

    File[] imageFiles = root.listFiles(pngFilter);

    MatVector images = new MatVector(imageFiles.length);

    int[] labels = new int[imageFiles.length];

    int counter = 0;
    int label;

    IplImage img=null;
    IplImage grayImg;

    int i1=mPath.length();

    for (File image : imageFiles) {
        String p = image.getAbsolutePath();
        img = cvLoadImage(p);

        if (img==null)
            Log.e("Error", "Error cvLoadImage");
        Log.i("image", p);

        int i2=p.lastIndexOf("-");
        int i3=p.lastIndexOf(".");
        int icount=Integer.parseInt(p.substring(i2+1,i3));
        if (count<icount) count++;

        String description=p.substring(i1,i2);

        if (labelsFile.get(description)<0)
            labelsFile.add(description, labelsFile.max()+1);

        label = labelsFile.get(description);

        grayImg = IplImage.create(img.width(), img.height(),
IPL_DEPTH_8U, 1);
    }
}

```

```

        cvCvtColor(img, grayImg, CV_BGR2GRAY);

        images.put(counter, grayImg);

        labels[counter] = label;

        counter++;
    }
    if (counter>0)
        if (labelsFile.max()>1)
            faceRecognizer.train(images, labels);
        labelsFile.Save();
    return true;
}

```

Figura 63. Código de entrenamiento de rostros.

Código de reconocimiento de rostros

Para el reconocimiento de rostros se usa el algoritmo LBPH de OpenCV el que realiza la comparación de las fotos del usuario almacenadas previamente, con los fotogramas que captura la cámara del dispositivo móvil Android, el código se muestra en la Fig. 64.

```

PersonRecognizer(String path){
    faceRecognizer =
com.googlecode.javacv.cpp.opencv_contrib.createLBPHFaceRecognizer(2,8,
8,8,200);
    mPath=path;
    labelsFile= new labels(mPath);
}

public String predict(Mat m) {
    if (!canPredict())
        return "";
    int n[] = new int[1];
    double p[] = new double[1];
    IplImage ipl = MatToIplImage(m,WIDTH, HEIGHT);
    faceRecognizer.predict(ipl, n, p);

    if (n[0]!=-1)
        mProb=(int)p[0];
    else
        mProb=-1;
}

```

```

        if (n[0] != -1)
            return labelsFile.get(n[0]);
        else
            return "Unkown";
    }

```

Figura 64. Código de reconocimiento de rostros.

4. PRUEBAS

Luego de terminado el desarrollo del código se procedió a realizar las pruebas que verifiquen el correcto funcionamiento de la aplicación móvil verificando que cumpla con los requerimientos propuestos al inicio.

4.1 Pruebas del sistema y arreglos

De acuerdo metodología Mobile-D se han procedido a realizar las pruebas siguientes pruebas: pruebas unitarias, pruebas de interfaz de usuario y pruebas funcionales, además de una lista de dispositivos con los que la aplicación móvil es compatible.

4.1.1 Pruebas Unitarias

Las pruebas unitarias se realizaron sobre el conjunto de clases y métodos que tienen funciones de comunicación, cifrado y reconocimiento facial.

4.1.1.1 Caso de Prueba 1: Guardar dispositivo móvil.

La realización de esta prueba se hizo para comprobar que en la configuración inicial se guarde los datos del dispositivo móvil donde se ha instalado la aplicación móvil. (Ver Tabla XXXIII)

TABLA XXXIII. PRUEBA UNITARIA: GUARDAR DISPOSITIVO MÓVIL.

Código	Nombre
PU001	Guardar dispositivo móvil.
Objetivo	Guardar los datos del dispositivo móvil dentro del dispositivo Arduino.

Pasos	<ol style="list-style-type: none"> 1. Inicializar la actividad. 2. Invocar el método agregarDispositivo. 3. Comprobar que el dispositivo móvil se haya guardado.
Resultados Esperados	El método debe devolver un resultado numérico, no nulo.
Resultados Obtenidos	El métodos devuelve un valor numérico no nulo.

El código utilizado para la elaboración de la prueba está descrita a continuación en la Fig. 65:

```
@Test
public void testAgregarDispositivoMóvil () {
    assertNotNull (detector.agregarDispositivo ("40f3d08a8add234ec4cd24a6
bb893b9", "0"));
}
```

Figura 65. Prueba unitaria: Guardar dispositivo móvil

La correcta ejecución del método se presenta en la Figura 66:

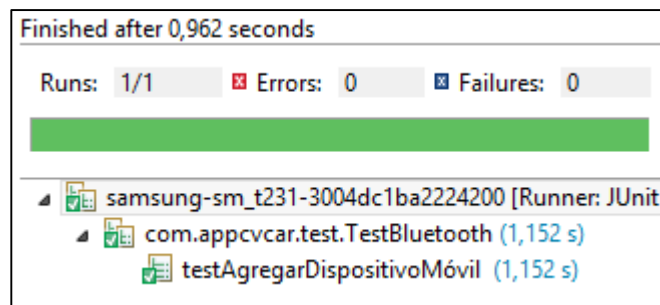


Figura 66 .Ejecución de Prueba Unitaria: Guardar dispositivo móvil

4.1.1.2 Caso de Prueba 2: Cifrar datos.

La realización de esta prueba se hizo para comprobar que los datos a enviarse a través de Bluetooth sean cifrados de manera correcta. (Ver Tabla XXXIV)

TABLA XXXIV. PRUEBA UNITARIA: CIFRAR DATOS.

Código	Nombre
PU002	Cifrar datos.
Objetivo	Cifrar los datos a enviarse a través de bluetooth.
Pasos	<ol style="list-style-type: none">1. Inicializar la actividad.2. Invocar el método cifrarDatos y colocar la información a ser cifrada.3. Comprobar que los resultados coincidan con los esperados.
Resultados Esperados	El método debe devolver una cadena cifrada igual a la esperada.
Resultados Obtenidos	El método debe devuelva una cadena cifrada igual a la esperada.

El código utilizado para la elaboración de la prueba está descrita a continuación en la Fig. 67:

```
public void testCifrarDatos() {  
    assertEquals(detector.cifrarDatos("desbloquear"),  
        "62e185e2b17bbl1a7563741248eeeff9f");  
}
```

Figura 67. Prueba unitaria: Cifrar datos

La correcta ejecución del método se presenta en la Fig.68:

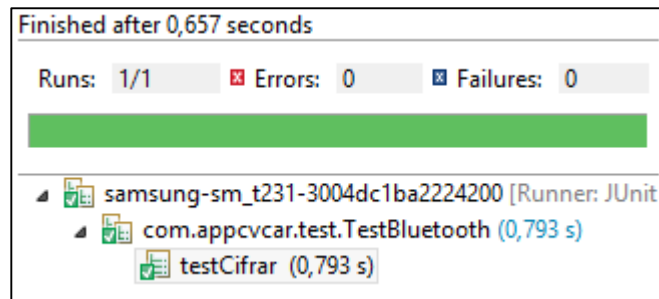


Figura 68 .Ejecución de Prueba Unitaria: Cifrar datos

4.1.1.3 Caso de Prueba 3: Enviar datos.

La realización de esta prueba se hizo para comprobar que los datos se envíen de manera correcta a través de bluetooth hacia el dispositivo Arduino (Ver Tabla XXXV)

TABLA XXXV. PRUEBA UNITARIA: ENVIAR DATOS A TRAVÉS DE BLUETOOTH.

Código	Nombre
PU003	Enviar datos.
Objetivo	Enviar los datos a través de Bluetooth.
Pasos	<ol style="list-style-type: none">1. Inicializar la actividad.2. Invocar el método cifrarDatos para cifrar los datos a enviarse.3. Invocar al metodo enviaDatos para enviar los datos que se cifraron anteriormente.4. Comprobar que se reciba una respuesta por parte del dispositivo <u>Arduino</u>.
Resultados Esperados	El método debe enviar los datos correctamente. El método debe devolver un resultado no nulo.
Resultados Obtenidos	El método ha enviado los datos correctamente. El método devuelve un resultado no nulo.

El código utilizado para la elaboración de la prueba está descrita a continuación en la Fig. 69:

```
@Test
public void testEnviarDatos() {
    assertNotNull(detector.enviarDatos(detector.cifrarDatos("block_car"), device));
}
```

Figura 69. Prueba unitaria: Enviar datos

La correcta ejecución del método se presenta en la Fig. 70:

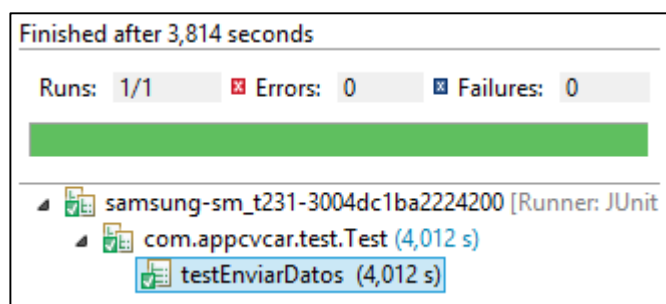


Figura 70 .Ejecución de Prueba Unitaria: Enviar datos

4.1.1.4 Caso de Prueba 4: Logueo facial.

La realización de esta prueba se hizo para comprobar que después del reconocimiento facial se realice el desbloqueo del dispositivo Arduino. (Ver Tabla XXXVI)

TABLA XXXVI. PRUEBA UNITARIA: GUARDAR DISPOSITIVO MÓVIL.

Código	Nombre
PU004	Logueo Facial.
Objetivo	Desbloquear el dispositivo Arduino después del reconocimiento facial.
Pasos	<ol style="list-style-type: none"> 1. Inicializar la actividad. 2. Invocar el método login con parámetros como son el identificador de usuario y dirección mac del dispositivo Arduino. 3. Comprobar que el dispositivo Arduino responda adecuadamente.
Resultados	El método debe enviar los datos correctamente.
Esperados	El método debe activar el dispositivo Arduino. El método debe devolver un resultado no nulo equivalente al esperado.
Resultados Obtenidos	El método ha enviado los datos correctamente. El método ha activado el dispositivo Arduino. El método devuelve un resultado no nulo equivalente al esperado.

El código utilizado para la elaboración de la prueba está descrita a continuación en la Fig. 71:


```
@Test
public void loginFacial() {
    assertEquals(detector.login("jinsop", "10:14:07:01:09:86"),
"21232f297a57a5a743894a0e4a801fc3");
}
```

Figura 71. Prueba unitaria: Logueo facial

La correcta ejecución del método se presenta en la Fig. 72:

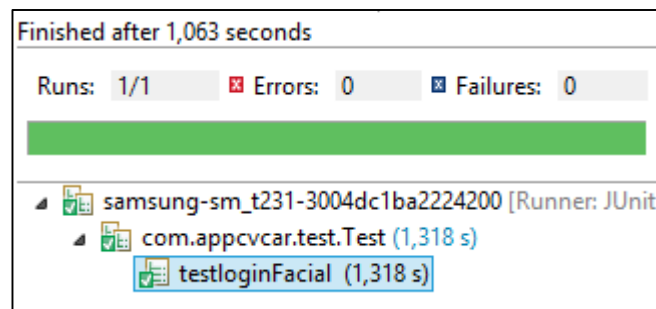


Figura 72 .Ejecución de Prueba Unitaria: Logueo Facial

4.1.1.5 Caso de Prueba 5: Logueo usuario.

La realización de esta prueba se hizo para comprobar que después del ingreso de la cédula y la clave se realice la verificación de datos y el desbloqueo del dispositivo Arduino. (Ver Tabla XXXVII)

TABLA XXXVII. PRUEBA UNITARIA: LOGUEO USUARIO.

Código	Nombre
PU005	Logueo Usuario.
Objetivo	Desbloquear el dispositivo Arduino después de la verificación de la cédula y clave ingresadas.
Pasos	<ol style="list-style-type: none"> 1. Inicializar la actividad. 2. Invocar el método loginClave con parámetros como son la cédula, clave de usuario y dirección mac del dispositivo Arduino. 3. Comprobar que el dispositivo Arduino responda

Resultados Esperados	El método debe enviar los datos correctamente. El método debe activar el dispositivo Arduino. El métodos debe devolver un resultado no nulo equivalente al
Resultados Obtenidos	El método ha enviado los datos correctamente. El método ha activado el dispositivo Arduino. El métodos devuelve un resultado no nulo equivalente al esperado.

El código utilizado para la elaboración de la prueba está descrita a continuación en la Fig. 73:

```
@Test
public void loginClave(){
    assertEquals(detector.loginClave("1105432163","12345678",
"10:14:07:01:09:86"), "21232f297a57a5a743894a0e4a801fc3");
}
```

Figura 73. Prueba unitaria: Logueo clave

La correcta ejecución del método se presenta en la Fig. 74:

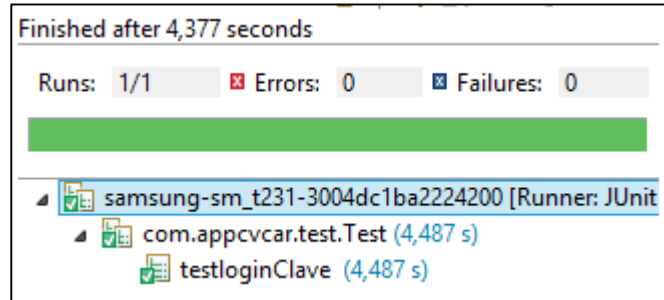


Figura 74 .Ejecución de Prueba Unitaria: Logueo Facial

4.1.2 Pruebas de interfaz de Usuario

Con este tipo de pruebas lo que se busca es comprobar que el diseño y conexión de los prototipos de pantalla es igual al obtenido en la aplicación desarrollada. Además también ayudara a verificar el control sobre el ingreso de datos.

4.1.2.1 Verificación de pantallas

A continuación se describe la tabla XXXVIII en la que se puede verificar que la aplicación posee las mismas pantallas que el Prototipado de pantallas.

TABLA XXXVIII. VERIFICACIÓN DE PANTALLAS.

Prototipos de Pantalla	Pantallas de Aplicación	Cumplimiento
Splash	Splash	✓
Asistente de configuración	Asistente de configuración	✓
Dispositivos Arduino	Dispositivos Arduino	✓
Detector Bluetooth	Detector Bluetooth	✓
Instrucciones entrenamiento facial	Instrucciones entrenamiento facial	✓
Entrenamiento facial	Entrenamiento facial	✓
Datos personales	Datos personales	✓
Finalizar configuración	Finalizar configuración	✓
Logeo facial	Logeo facial	✓
Dispositivos Arduino	Dispositivos Arduino	✓
Logeo clave	Logeo clave	✓
Administrador	Administrador	✓
Administrar usuarios	Administrar usuarios	✓
Dispositivos móviles	Dispositivos móviles	✓
Rostros de usuarios	Rostros de usuarios	✓
Ajustes de aplicación	Ajustes de aplicación	✓

Usuario	Usuario	✓
---------	---------	---

4.1.2.2 Comprobación de Datos Ingresados

Para comprobar que la aplicación realiza un adecuado control sobre la información ingresada desarrollaron pruebas específicas sobre las pantallas que poseen entrada de datos.

En la aplicación la pantalla que poseen un ingreso de datos son: Datos personales y Nuevo usuario.

PIU01: Datos personales

En esta pantalla se tiene como objetivo ingresar los datos personales del usuario administrador como describe la tabla XXXIX.

TABLA XXXIX. COMPROBACIÓN DATOS INGRESADOS: DATOS PERSONALES.

Código	Nombre
PIU01	Control de datos ingresados a la pantalla Datos personales
Objetivo	Verificar que los datos ingresados a la pantalla Datos personales cumplan con el formato requerido.
Casos	<ol style="list-style-type: none"> 1. Ingresar los textos numéricos y letras de acuerdo al campo requerido y dejar el campo nombres vacío. 2. Ingresar los textos numéricos y letras de acuerdo al campo requerido e ingresar una contraseña con menos de 8 caracteres. 3. Ingresar los textos numéricos y letras de acuerdo al campo requerido e ingresar las contraseñas diferentes.

Resultados Esperados	<p>La aplicación debe funcionar correctamente cuando el usuario ingrese una cadena de texto con letras y/o números.</p> <p>En caso de que las contraseñas no coincidan debe presentar un mensaje.</p> <p>En caso de que el texto esté vacío, la aplicación debe presentar un mensaje de información.</p>
Resultados Obtenidos	<p>La aplicación funciona correctamente cuando se ingresan letras y números de acuerdo al campo requerido.</p> <p>La aplicación presenta los mensajes en caso de campos vacíos o donde las contraseñas no coincidan o el tamaño no es igual o mayor a 8 caracteres.</p>

La ejecución de la prueba realizada se muestra en la Fig.75, Fig. 76 y la Fig. 77.

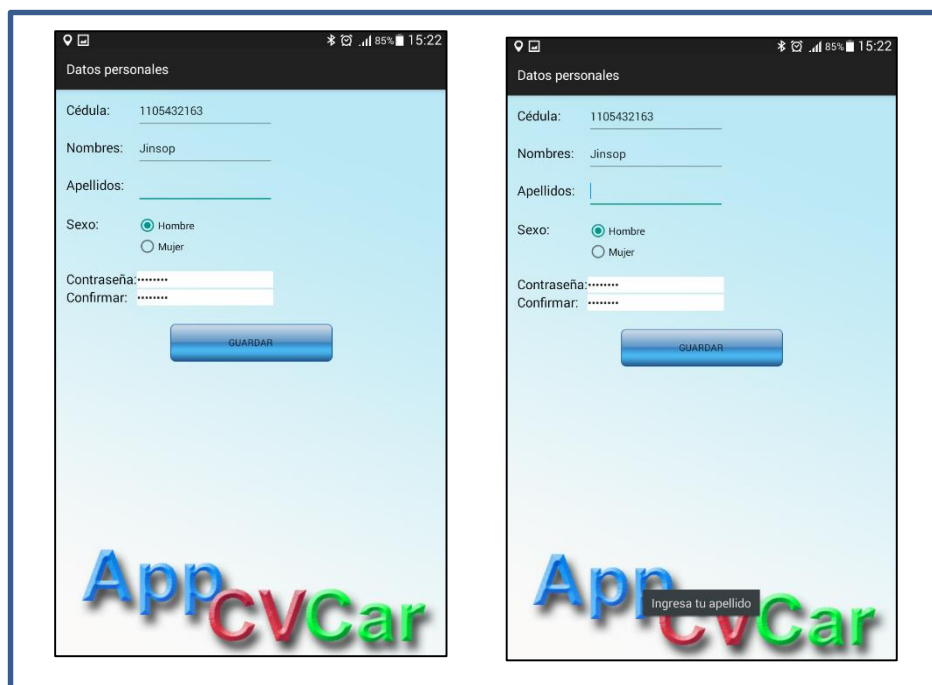


Figura 75. Comprobación de datos ingresados: Datos personales campo vacío.

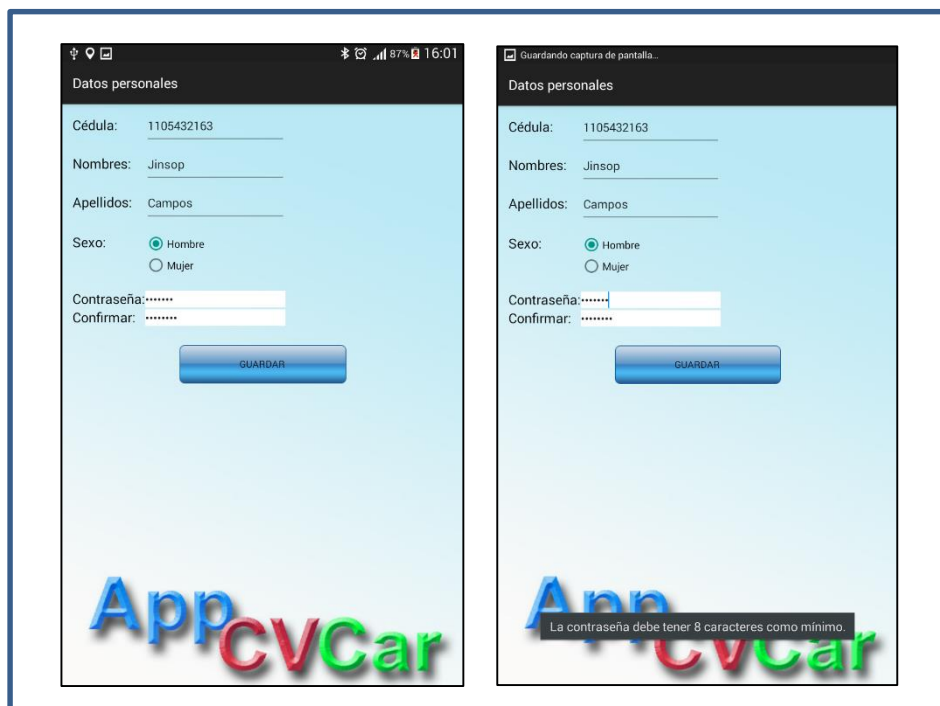


Figura 76. Comprobación de datos ingresados: Datos personales contraseña menor a 8 caracteres

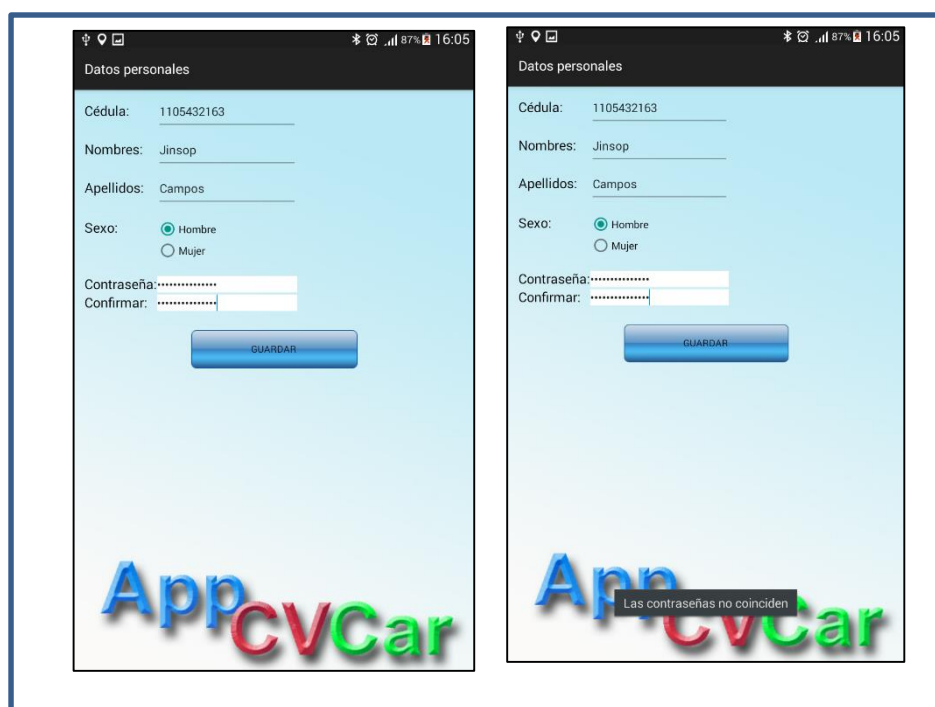


Figura 77. Comprobación de datos ingresados: Datos personales contraseñas diferentes.

4.1.3 Pruebas funcionales

En base a los requerimientos planteados en la etapa de Análisis se procedió a verificar el cumplimiento de cada uno de ellos, como se puede ver en la tabla XL.

TABLA XL. VERIFICACIÓN DE REQUERIMIENTOS.

Código	Descripción	Categoría	Cumplimiento
RF001	Al usuario detectar los dispositivos Bluetooth.	EVIDENTE	✓
RF002	Al usuario conectarse al dispositivo electrónico Arduino una vez que se haya terminado el proceso de detección.	EVIDENTE	✓
RF003	Al usuario verificar si el dispositivo se encuentra configurado o si el dispositivo se encuentra en estado funcional.	EVIDENTE	✓
RF004	Al usuario poder iniciar con el entrenamiento de su patrón facial a través de las cámaras que posea el dispositivo móvil para su posterior identificación.	EVIDENTE	✓
RF005	Al usuario guardar los datos del patrón facial obtenidos en el entrenamiento dentro del dispositivo Android.	EVIDENTE	✓
RF006	Al usuario obtener y almacenar los datos personales del usuario dentro del dispositivo electrónico Arduino.	EVIDENTE	✓
RF007	Al usuario guardar el nombre y la dirección MAC del dispositivo dentro del dispositivo electrónico Arduino.	EVIDENTE	✓
RF008	Al usuario administrar los dispositivos Bluetooth almacenados dentro del dispositivo electrónico Arduino.	EVIDENTE	✓

RF009	Al usuario administrar sus datos personales almacenados dentro del dispositivo electrónico Arduino.	EVIDENTE	✓
RF010	Al usuario poder usar cualquiera de las cámaras que posea el dispositivo móvil.	EVIDENTE	✓
RF011	Al usuario poder detectar su rostro al enfocar su cámara.	EVIDENTE	✓
RF012	Al usuario verificar su identidad con los datos previamente guardados en el entrenamiento dentro del dispositivo electrónico Arduino.	EVIDENTE	✓
RF013	Al usuario desbloquear el dispositivo electrónico Arduino, aún cuando no existan las condiciones de luz necesarias para el reconocimiento facial mediante otro mecanismo seguro.	EVIDENTE	✓
RF014	Al usuario administrar los rostros almacenados dentro del dispositivo móvil.	EVIDENTE	✓
RF015	Al usuario permitir y denegar las conexiones entrantes hacia el dispositivo electrónico Arduino.	EVIDENTE	✓
RF016	Al usuario permitir reestablecer las configuraciones iniciales de la aplicación móvil y del dispositivo electrónico en caso de que así lo requiera.	EVIDENTE	✓

4.1.3.1 PF01: Configuración inicial.

TABLA XLI. PRUEBA FUNCIONAL: CONFIGURACIÓN INICIAL.

Código	Nombre
PF001	Prueba funcional de la configuración inicial
Objetivo	Comprobar el proceso de configuración inicial se realice correctamente
Pasos	<ol style="list-style-type: none">1. El usuario abre la aplicación.2. Se presenta la pantalla de Bienvenida al Asistente de configuración.3. El usuario presiona en el botón Empezar.4. Se presenta la pantalla Dispositivos Bluetooth con todos los dispositivos detectados.5. El usuario selecciona el dispositivo Arduino a configurar.6. Se presenta la pantalla Instrucciones Entrenamiento facial con indicaciones para el entrenamiento de rostro.7. El usuario presiona en el botón Comenzar.8. Aparece la pantalla donde muestra la cámara.9. El usuario enfoca la cámara sobre su rostro.10. El usuario presiona sobre el botón entrenar y escribe su nombre.11. El usuario presiona en el botón grabar y se graba su rostro.12. Aparece la pantalla Datos personales.13. El usuario ingresa sus datos personales y presiona en el botón Guardar.14. Aparece la pantalla de configuración finalizadas y el usuario presiona en el botón Finalizar.15. Termina la configuración de la aplicación móvil y el dispositivo Arduino.
Resultados Esperados	<p>Correcta conexión y transmisión de datos por bluetooth.</p> <p>Almacenamiento de 30 fotos del rostro del usuario.</p> <p>Almacenamiento de datos personales del usuario.</p> <p>Almacenamiento de datos del dispositivo móvil Android.</p>

	Correcta finalización del asistente de configuración.
Resultados Obtenidos	<p>Se ha obtenido una correcta conexión y transmisión de datos por bluetooth.</p> <p>Se han almacenado correctamente 30 fotos del rostro del usuario.</p> <p>Se ha almacenado los datos personales del usuario.</p> <p>Se ha almacenado correctamente de datos del dispositivo móvil Android.</p> <p>Se ha conseguido una correcta finalización del asistente de configuración.</p>

La ejecución realizada se muestra en la Fig. 78.



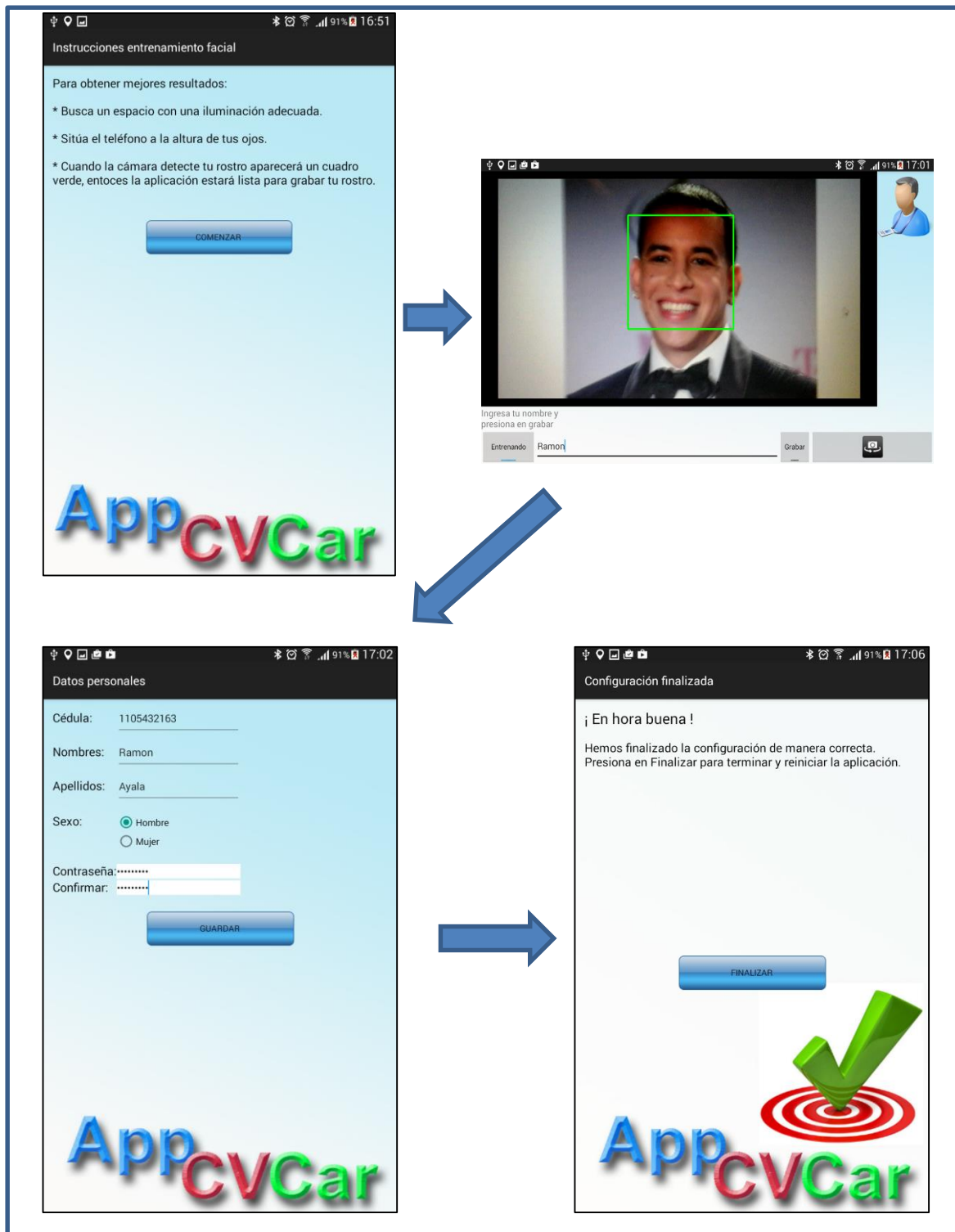


Figura 78. Prueba Funcional: Configuración Inicial.

4.1.3.2 PF02: Logueo facial.

TABLA XLII. PRUEBA FUNCIONAL: LOGUEO CLAVE.

Código	Nombre
PF002	Prueba funcional de la Logueo facial
Objetivo	Comprobar el acceso a la aplicación móvil a través de reconocimiento facial.
Pasos	<ol style="list-style-type: none">1. El usuario abre la aplicación2. Se presenta la pantalla de Dispositivos Arduino.3. El usuario elige el dispositivo a usar y presiona la opción desbloquear.4. Se presenta la pantalla de Logueo facial.5. El usuario enfoca la cámara sobre su rostro y presiona el botón Ingresar.6. La aplicación móvil detecta, reconoce el rostro y envía los datos para su verificación en el dispositivo Arduino.7. El dispositivo arduino verificar los datos, verificar que tipo de usuario y desbloquea el encendido y seguros del vehículo.8. De acuerdo al tipo de usuario muestra la pantalla con las diferentes opciones de acuerdo al rol que desempeña.
Resultados Esperados	<p>Correcta conexión y transmisión de datos por bluetooth.</p> <p>Detección y reconocimiento del rostro del usuario.</p> <p>Verificación de datos personales del usuario.</p> <p>Desbloqueo del dispositivo Arduino si el usuario está autorizado.</p>
Resultados Obtenidos	<p>Se ha obtenido una correcta conexión y transmisión de datos por bluetooth.</p> <p>Se ha detectado y reconocido el rostro del usuario.</p> <p>Se ha Verificado correctamente los datos personales del usuario.</p> <p>Se ha desbloqueado del dispositivo Arduino con el usuario autorizado.</p>

Los resultados de la ejecución se muestra en la Fig. 79.

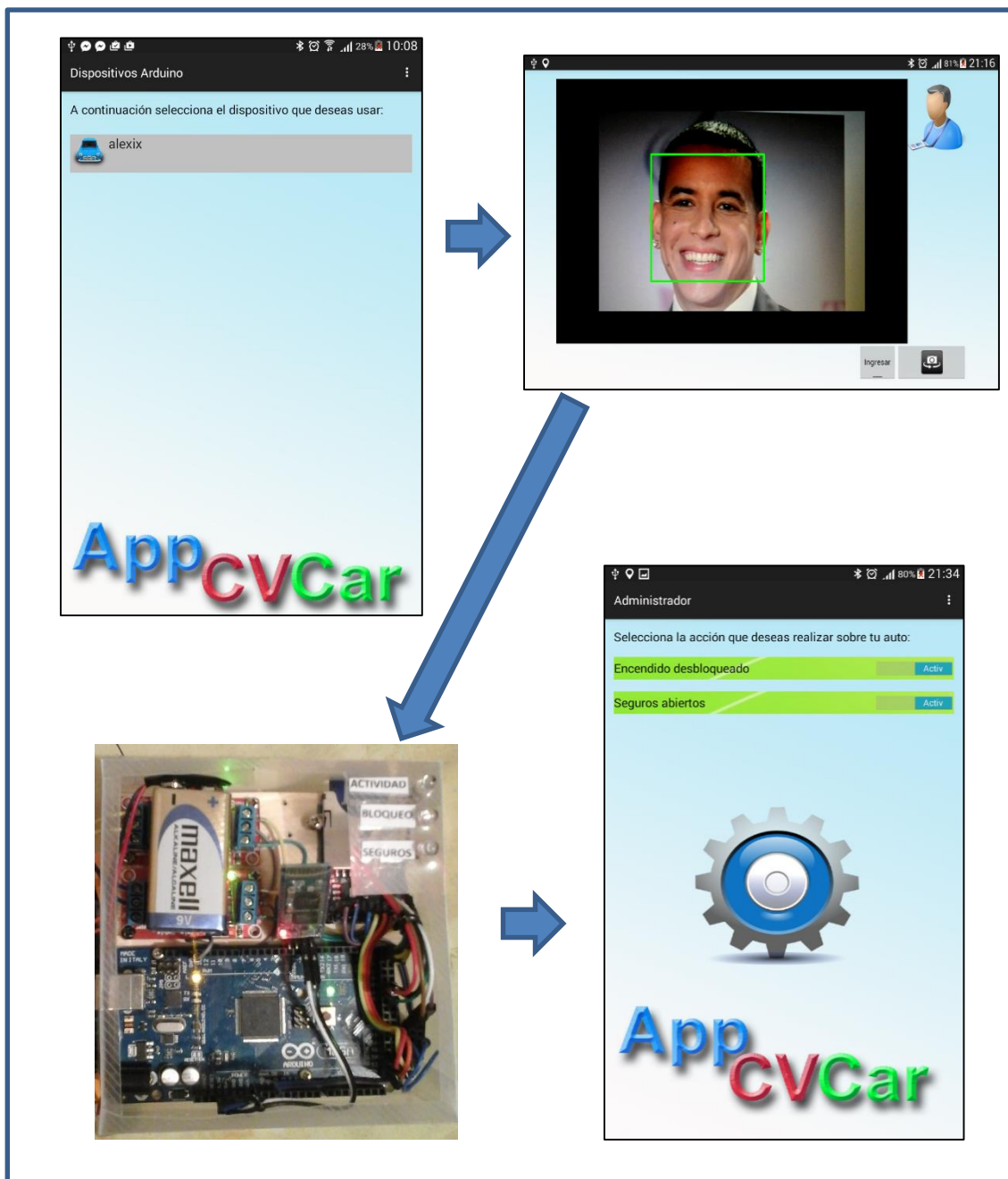


Figura 79. Prueba funcional: Logueo Facial.

4.1.3.3 PF03: Logueo clave.

TABLA XLIII. PRUEBA FUNCIONAL: LOGUEO CLAVE.

Código	Nombre
PF003	Prueba funcional del Logueo con clave
Objetivo	Comprobar el acceso a la aplicación móvil a través de clave.
Pasos	<ol style="list-style-type: none">1. El usuario abre la aplicación.2. En caso de que no se pueda reconocer el rostro por falta de iluminación, aparecerá un botón en la parte inferior de la pantalla con el nombre Ingresar.3. El usuario presiona en el botón Ingresar y se presenta la pantalla de Logueo clave.4. El usuario ingresa su clave y presiona en el botón Ingresar.5. La aplicación móvil detecta envía los datos para su verificación en el dispositivo Arduino.6. El dispositivo arduino verificar los datos, verificar que tipo de usuario y desbloquea el encendido y seguros del vehículo.7. De acuerdo al tipo de usuario muestra la pantalla con las diferentes opciones de acuerdo al rol que desempeña.
Resultados Esperados	Correcta conexión y transmisión de datos por bluetooth. Verificación de clave del usuario. Desbloqueo del dispositivo Arduino si el usuario está autorizado.
Resultados Obtenidos	Se ha obtenido una correcta conexión y transmisión de datos por bluetooth. Se ha verificado correctamente la clave del usuario. Se ha desbloqueado del dispositivo Arduino con el usuario autorizado.

El resultado de la ejecución se muestra en la Fig. 80.

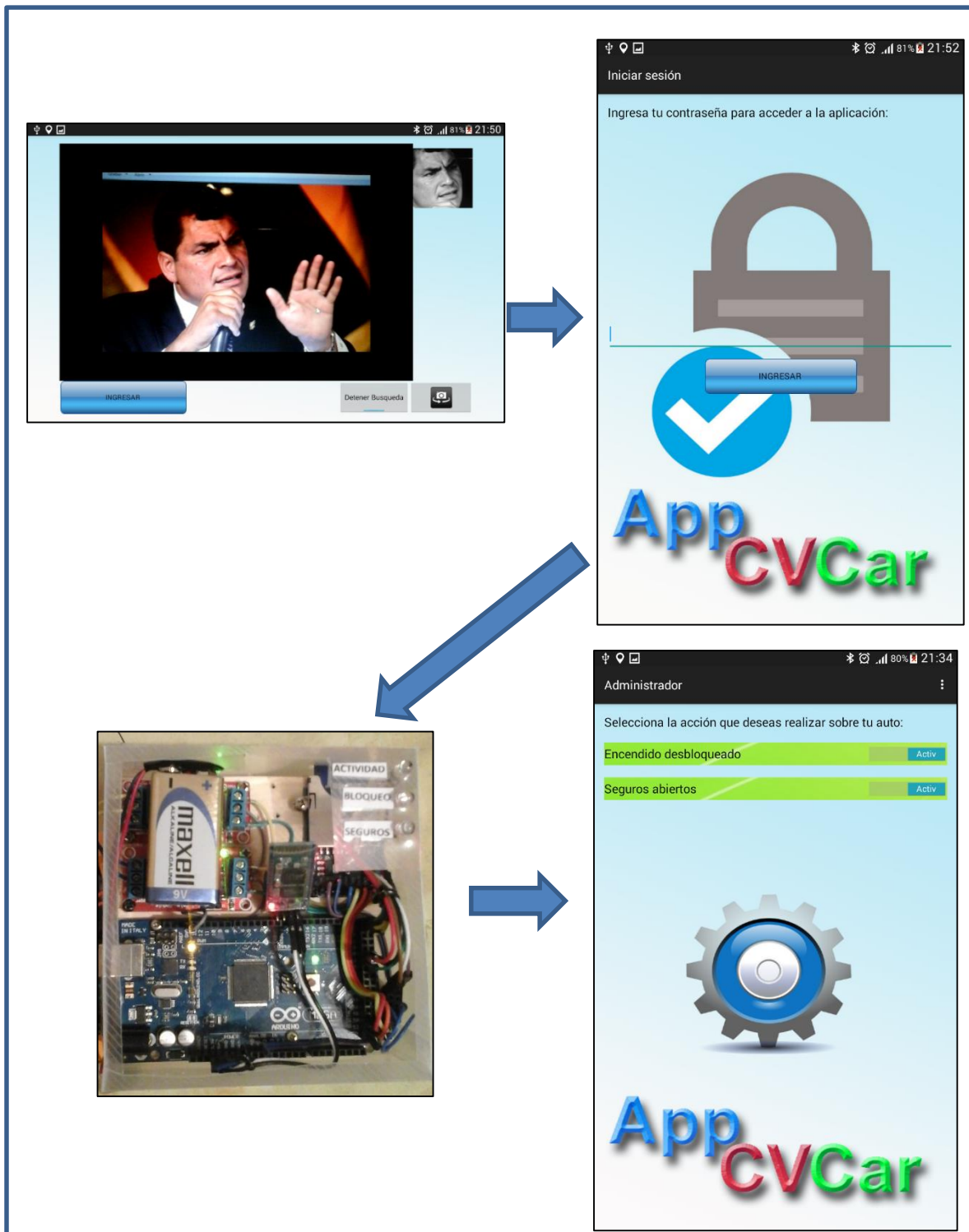


Figura 80. Prueba funcional: Logueo clave.

4.1.4 Dispositivos compatibles

Para comprobar que la aplicación se ejecuta correctamente en varios dispositivos, se realizó una prueba con la ayuda de la herramienta online apkudo la cual permite obtener una lista de los dispositivos móviles en los cuales una aplicación funciona sin ningún contratiempo. En la tabla XLIV se presentan los dispositivos en los que la aplicación funcionó adecuadamente.

TABLA XLIV. DISPOSITIVOS COMPATIBLES.

Marca	Nombre	Modelo	Versión de SO Android
HTC	One X+	One X+	4.1.1
	One X LTE	PJ83100	4.0.3
	One S	HTC One S	4.0.4
	One X	S720E	4.0.4
LG	Mach	LG-LS860	4.0.1
	L5	E610	4.0.3
	Intuition	VS950 4G	4.1.2
	G2	VS980	4.4.2
	Nexus 4	E960	4.4.4
Motorola	Photon 4G LTE	XT897	4.0.4
	Droid Razr Maxx HD	XT926	4.4.2
	Moto X (2013)	XT1053	4.4.4
Samsung	Galaxy Tab 4	SM-T231	4.4.4
	Galaxy Express	SGH-I437	4.0.4
	Galaxy SIII	SGH-I747	4.1.1
	Galaxy Reverb	SPH-M925	4.1.2
	Galaxy S4	SPH-L720T	4.4.2
	Galaxy S5	SM-G900	4.4.2
	Galaxy Note II	SGH-I317	4.3

g. DISCUSIÓN

En el siguiente apartado se muestra el análisis acerca de los resultados obtenidos con el desarrollo del presente trabajo de titulación.

1. Desarrollo de la propuesta alternativa.

1.1 Objetivo Específico 1: Determinar los requerimientos técnicos de Hardware y Software para la implementación del mecanismo de seguridad.

Para llevar a cabo este objetivo se realizó un análisis del software a emplearse, en este caso se empleó Android (Ver apartado 3.1.1 Android) como plataforma móvil, SQLite (Ver apartado 3.1.2 SQLite) como base de datos, OpenCV Android (Ver apartado 3.1.3 OpenCV Android) como herramienta para el reconocimiento facial, NDK Android (Ver apartado 3.1.4 NDK Android) para usar partes de código nativo C/C++, Eclipse IDE (Ver apartado 3.1.5 Eclipse IDE) como entorno de desarrollo para aplicaciones Android, Fritzing (Ver apartado 3.1.6 Fritzing) para diseño y simulación de esquemas electrónicos, y Arduino software (Ver apartado 3.1.7 Arduino software); para llevar a cabo el desarrollo de la aplicación móvil Android con reconocimiento facial.

De la misma forma se realizó un estudio y selección de los elementos hardware necesarios como son Arduino Mega 2560 (Ver apartado 3.2.2 Arduino Mega 2560) el cual sirvió para el procesamiento de datos, Módulo Bluetooth HC-06 (Ver apartado 3.2.3 Módulo Bluetooth HC-06) para el envío y recepción de datos de forma inalámbrica, Módulo lector de tarjetas SD (Ver apartado 3.2.4 Módulo Lector SD) para el almacenamiento de datos compartidos entre la aplicación móvil y el Arduino, y por último Módulos Relay (Ver apartado 3.2.5 Módulo Relay o Relé) para abrir y cerrar circuitos para la construcción del dispositivo electrónico que será el encargado de proteger el encendido del vehículo donde se vaya a instalar, para facilitar esta labor se seleccionó la plataforma de hardware y software Arduino, la cual es una excelente opción para trabajar con electrónica si necesitar tener un gran conocimiento dentro de este campo.

1.2 Objetivo Específico 2: Desarrollar la aplicación móvil android para detección de personas autorizadas.

Para llevar a cabo este objetivo se realizó todo el proceso basándose en la metodología de desarrollo de aplicaciones móviles Mobile-D (Ver sección 1. Métodos). En la fase de análisis se determinaron todos los requerimientos (Ver apartado 1.1.2 Requerimientos Iniciales, 1.1.2.1 Requerimientos Funcionales y 1.1.2.2 Requerimientos no funcionales), procesos (Ver apartado 1.1.2.3 Análisis de requerimientos), limitaciones (Ver apartado 1.1.3.2 Limitaciones), supuestos y dependencias (Ver apartado 1.1.3.3. Limitaciones y dependencias) además de los de los recursos software y hardware necesarios para desarrollar la aplicación (Ver apartado 1.1.4 Establecimiento del proyecto).

En la fase de diseño se realizó la configuración del ambiente de desarrollo (Ver apartado 2.1.1 Configuración del Ambiente de desarrollo) para poder desarrollar aplicaciones móviles Android con Visión artificial, planificación de fases de la metodología de desarrollo Mobile-D (Ver apartado 2.1.5.2 Planificación de fases), también se ha diseñado la arquitectura funcionamiento del prototipo (Ver figura 32), construcción del modelo del dominio (Ver figura 33), el diseño de base de datos (Ver figura 34), de la misma forma se ha incluido la navegabilidad de la aplicación móvil (Ver figura 35), sus pantallas y el storycard de cada una de ellas (Ver apartado 2.1.6.3 Descripción de la Interfaz de Usuario), todos estos elementos necesario en la arquitectura de la aplicación.

Después en la etapa de codificación se ha considerado los estándares de codificación (Ver apartado 3.1.1. Estándares de codificación), estructura de directorios (Ver apartado 3.1.2) además de ciertas partes de código desarrollado (Ver apartado 3.1.3).

Todas las funciones han sido debidamente probadas para garantizar un correcto funcionamiento de la aplicación móvil y la interacción con el dispositivo electrónico Arduino.

1.3 Objetivo Específico 3: Construir un prototipo de aplicación móvil android de reconocimiento facial conectada a la placa arduino para proteger el sistema de encendido del vehículo.

Para cumplir con este objetivo lo que se ha realizado es la construcción del dispositivo electrónico Arduino el cual será el que actuará como un protector del encendido del vehículo, con lo cual una vez terminado se procede a integrarlo con la aplicación móvil, dando forma al prototipo propuesto en el presente proyecto de titulación.

Para el diseño y construcción del dispositivo Arduino se ha tomado en cuenta los materiales hardware (Ver apartado 2.1.7.1 Materiales Hardware), así como también como los esquemas y pines de conexión (Ver apartado 2.1.7.2 Esquemas de conexión), esquema final de conexiones (ver apartado 2.1.7.3 Esquema final de conexiones).

Estas dos partes interactuaran entre sí a través de comunicación inalámbrica bluetooth, hay que destacar que la aplicación móvil y el dispositivo electrónico Arduino son dependientes entre sí, ya que trabajan con los datos que se almacenen dentro de la memoria del dispositivo electrónico ya mencionado, ninguno de los dos pueden funcionar de forma independiente.

Se han realizado las pruebas correspondientes utilizando tanto la aplicación móvil como también el Dispositivo Arduino, entre las pruebas que se han considerado tenemos las pruebas Unitarias (Ver apartado 4.1.1 Pruebas Unitarias), pruebas de interfaz de usuario (Ver apartado 4.1.2. Pruebas de Interfaz de Usuario), pruebas funcionales (Ver apartado 4.1.3 Pruebas funcionales), y por último los dispositivos con los que la aplicación móvil es compatible (Ver apartado 4.1.4 Dispositivos compatibles), todo esto ha permitido verificar el correcto funcionamiento del prototipo construido.

h. CONCLUSIONES

- El uso de la plataforma Arduino es una buena opción al momento de trabajar con electrónica y en proyectos de automatización, ya que no se requiere tener un alto nivel de conocimientos en el tema, además de que se convierte en una experiencia interesante al trabajar con este tipo de proyectos.
- La Implementación de librerías como OpenCV simplifica de manera significativa el desarrollo de aplicaciones móviles que requieran de funciones que utilizan una cámara como detección de rostros, reconocimiento de rostros, detectores de movimiento, detectores de objetos etc.
- La aplicación móvil ha sido desarrollada específicamente para la plataforma Android ya que al trabajar con OpenCV se necesita usar librerías propias del sistema operativo para obtener un mejor rendimiento en el procesamiento de imágenes utilizadas para el reconocimiento facial.
- La visión artificial es un campo aún en desarrollo por lo que no se pueden esperar resultados muy exactos, ya que su funcionamiento se limita a la tecnología de cámaras disponibles y de las condiciones de iluminación del ambiente en donde se trabaje.
- La metodología de desarrollo de móviles Mobile-D ha resultado muy útil en el presente trabajo de titulación ya que ha permitido un desarrollo ágil considerando que solo se ha contado con un solo miembro de equipo de desarrollo en este caso el tesista.

i. RECOMENDACIONES

Una vez finalizadas todas las etapas del presente proyecto de titulación, se ha creído conveniente realizar las siguientes recomendaciones:

- Para trabajar con reconocimiento facial se recomienda el uso de OpenCV ya que ofrece resultados muy buenos además de ser de código abierto, contar con un amplio uso en la industria y además de contar con el debido soporte en la web.
- Se recomienda utilizar el trabajo realizado como base para futuros proyectos relacionados con la temática, pudiendo ampliar sus funcionalidades usando el hardware proporcionado por Arduino ampliando su alcance como podría ser a través de wifi o también a través de las redes móviles de datos.
- Para la manipulación de dispositivos electrónicos tener precauciones ya que son elementos delicados que pueden estropearse o convertirse en elementos peligrosos si no se les da el debido manejo.
- Trabajar en proyectos relacionados con software y hardware ya que brinda una gran experiencia y aprendizaje, potenciando las habilidades del futuro profesional para el desarrollo de proyectos tecnológicos e innovadores

j. BIBLIOGRAFÍA

- [1] Y. Daniel and Y. D. Amaya Balaguera, "Metodologías ágiles en el desarrollo de aplicaciones," *Journal Technology*, vol. 12, no. 2, p. 14.
- [2] G. H. Forman and J. Zahorjan, *The Challenges of Mobile Computing*, IEEE Computer, 1994.
- [3] O. Déniz, J. Salido and G. Bueno, *Programación de Apps de Visión Artificial*, Bubok Publishing, 2013.
- [4] R. M. Correa, *Composición del sistema operativo móvil iOS de Apple y el hardware y software que lo utilizan.*, 2014.
- [5] R. C. Maneiro, *Las mejores aplicaciones iOS para iPad y iPhone de Apple.*, 2011.
- [6] K. M. Polanco and J. L. Taibo, *Android el sistema operativo de Google para dispositivos móviles*, 2011.
- [7] K. WorldPanel, "Android maintains its European domination," septiembre 2011. [Online]. Available: <http://www.kantarworldpanel.com/Global/News/Android-maintains-its-European-domination>.
- [8] J. T. Girónes, *El gran libro de Android*, Marcombo, 2012.
- [9] J. Yeray, *Windows Phone 7.5 Desarrollo de aplicaciones en Silverlight*, 2011.
- [10] H. Ramírez García, *Análisis de sistemas operativos smartphone.*, 2014.
- [11] E. M. García, *Visión Artificial*, Catalunya: Universitat Oberta de Catalunya.
- [12] G. D. I. C. Pajares, *Visión por computador. Imágenes Digitales y Aplicaciones*, Rama, 2001.
- [13] H. Rodríguez, *Imagen digital: conceptos básicos.*, Marcombo, 2009.
- [14] K. S. Klette and A. . K. Reinhard, *Computer vision: three-dimensional data from images*, Singapore, 2001.
- [15] D. A. Gómez Allende, *Reconocimiento de formas y visión artificial*, 1993.
- [16] P. Roldán Ruz, *Visión por computador en iPhone 4*, Universitat Oberta de Catalunya, 2012.

- [17] J. A. Espallargas, Guia turistica de monumentos basada en reconocimiento visual con un móvil, Zaragoza: Universidad de Zaragoza, 2012.
- [18] G. Bradsky, The OpenCV Library.Dr. Dobb's Journal of Software Tools, 2000.
- [19] O. D. Team., "OpenCV4Android SDK," [Online]. Available: http://docs.opencv.org/doc/tutorials/introduction/android_binary_package/O4A_SDK.html.
- [20] J. Fernández Martí, Desarrollo de una aplicación Android para la detección de señales de tráfico usando OpenCV, 2015.
- [21] O. Developers, "OpenCV IOS," [Online]. Available: http://docs.opencv.org/2.4/doc/tutorials/ios/video_processing/video_processing.html.
- [22] F. CHICHIZOLA, A. E. DE GIUSTI and M. NAIIOUF, "Reconocimiento de rostros," En IV Workshop de Investigadores en Ciencias de la Computación, 2002.
- [23] P. N. J. P. H. a. D. J. K. Belhumeur, "Recognition using class specific linear projection. Pattern Analysis and Machine Intelligence," IEEE Transactions on, 1997, vol. 19, no 7, p. 711-720..
- [24] E. A. CORTÉS, M. G. MARTÍNEZ and N. G. RUBIO, "Linear discriminant analysis versus adaboost for failure forecasting," 2008.
- [25] Y. A. CAMA CASTILLO, "Prototipo computacional para la detección y clasificación de expresiones faciales mediante la extracción de patrones locales binarios.," 2015.
- [26] J. D. ALVARADO and J. FERNÁNDEZ, "Análisis de textura en imágenes a escala de grises, utilizando patrones locales binarios (LBP)," ENGI Revista Electrónica de la Facultad de Ingeniería, 2012, vol. 1, no 1., 2012.
- [27] J. B. Peña, Estudio de la plataforma Android, 2008.
- [28] SQLite, "About SQLite," [Online]. Available: <https://www.sqlite.org/about.html>. [Accessed 20 03 2015].
- [29] E. d. d. d. OpenCV, "OpenCV for Android SDK," [Online]. Available: <http://opencv.org/about.html>. [Accessed 03 03 2015].
- [30] A. Developers, "Android NDK," [Online]. Available: <https://developer.android.com/tools/sdk/ndk/index.html>.

- [31] D. Gallardo, "IBM DeveloperWorks," IBM, [Online]. Available: <http://www.ibm.com/developerworks/ssa/library/os-ecov/>. [Accessed 20 03 2015].
- [32] Fritzing, "Fritzing electronics made easy," [Online]. Available: <http://fritzing.org/home/>. [Accessed 13 03 2015].
- [33] Arduino, "Aduino," [Online]. Available: <http://arduino.cc/en/Main/ArduinoBoardMega2560>. [Accessed 22 03 2015].
- [34] S. Monk, 30 Proyectos con Arduino, Madrid: Estribor, 2012.
- [35] H. R. R. University, "Reutlingen University," [Online]. Available: http://www.tec.reutlingen-university.de/uploads/media/DatenblattHC-05_BT-Modul.pdf. [Accessed 23 03 2015].
- [36] TecBolivia, "Tecnología en Bolivia," [Online]. Available: <http://tecbolivia.com/index.php/venta-de-componentes-electronicos-11/shields-y-accesorios-arduino/placa-modulo-lector-sd-card-detail>. [Accessed 23 03 2015].
- [37] S. Electrónica, "SYSPORT Electrónica," [Online]. Available: <http://www.sysport.com.ar/modulo-rele-relay-de-canal-5v-10a-arduino-pic-avr-robotica-8227473xJM.html>. [Accessed 23 03 2015].
- [38] J. D. Amaya Balaguera, "Metodologías ágiles en el desarrollo de aplicaciones para dispositivos móviles," [Online]. Available: http://www.uelbosque.edu.co/sites/default/files/publicaciones/revistas/revista_tecnologia/volumen12_numero2/12Articulo_Rev-Tec-Num-2.pdf.
- [39] J. S. G. B. Oscar Déniz, Programación de Apps de Visión Artificial, España: Bubok Publishing S.L., 2013.
- [40] G. R. Acevedo, Ciencia, tecnología y sociedad: una mirada desde la Educación en Tecnología., Revista Iberoamericana de Educación, 1998.
- [41] J. L. De la Fuente Mínguez, La seguridad activa y pasiva en un vehículo, Inversiones Editoriales Dossat, 1995.
- [42] A. Riu, Sistemas eléctricos del automóvil, Ediciones Mundo Técnico, 1999.
- [43] C. O. Vélez Espinoza and W. Naranjo, Sistema antirrobo vehicular mediante mando a distancia del encendido eléctrico, 2007.
- [44] J. Gosling, The Java language specification, Addison-Wesley Professional., 2000.

- [45] R. Rogers, Android application development: Programming with the Google SDK, O'Reilly Media Inc., 2009.
- [46] J. O. Vera, INTRODUCCION AL PROCESAMIENTO DIGITAL DE IMAGENES. UN ENFOQUE TEORICO-PRACTICO UTILIZANDO OPEN CV., 2013.
- [47] A. developers, "Android NDK," [Online]. Available: <http://developer.android.com/intl/es/tools/sdk/ndk/index.html>. [Accessed 25 10 2015].
- [48] S. Monk, 30 proyectos con Arduino, Estribor, 2013.
- [49] F. developer, "Fritzing electronic made easy," [Online]. Available: <http://fritzing.org/home/>. [Accessed 25 10 2015].

k. ANEXOS

Anexo 1. Artículo científico

PROTOTIPO DE DETECCIÓN DE PERSONAS AUTORIZADAS PARA ENCENDER UN VEHÍCULO APLICANDO TÉCNICAS DE VISIÓN ARTIFICIAL EN DISPOSITIVOS MÓVILES ANDROID

J. Campos, M. Palma, H. Paz

Abstract— This paper is the result of having applied a set of software and hardware technologies for the realization of an Android mobile application detection authorized to access a particular vehicle through facial recognition users, for which it uses and manages a device electronic platform based on Arduino hardware and software will be installed in the car.

Keywords— Arduino, Visión Artificial, Reconocimiento facial, Android, Bluetooth.

Resumen: Este artículo es el resultado de haber aplicado un conjunto de tecnologías de software y hardware para la realización de una aplicación móvil Android de detección de usuarios autorizados para poder acceder a un vehículo determinado, a través de reconocimiento facial, para lo cual utiliza y administra un dispositivo electrónico basado en la plataforma de hardware y software Arduino, el mismo que se instalará en el interior de un automóvil donde se quiera probar su funcionamiento.

I. INTRODUCCIÓN

Los dispositivos móviles se han convertido en los últimos años, en una de las tecnologías más utilizadas a diario por millones de personas alrededor de mundo, lo que ha hecho a estos dispositivos convertirse en una parte fundamental en el desarrollo cotidiano de cualquier actividad que realice una persona. Dicho esto, también se han ido acoplando con otros campos como son los automóviles, diseño gráfico, negocios, etc, con el fin de mejorar los servicios que estos ofrecen.

Considerando lo anterior se ha tomado en cuenta los problemas de seguridad de la que disponen los vehículos o el uso que se le pueda dar en caso de que caigan en manos equivocadas, lo que hace necesario tener un mecanismo para protegerlo de accesos no autorizados a través de dispositivos móviles.

Por ello, se ha propuesto la realización del presente proyecto cuyo objetivo principal desarrollar un prototipo de aplicación móvil de visión artificial que brinde un mecanismo de seguridad a vehículos para restringir su uso por parte de personas no autorizadas mediante reconocimiento facial.

Para el cumplimiento del mismo se establecieron tres objetivos específicos, los cuales son: determinar los requerimientos técnicos de Hardware y Software para la implementación del mecanismo de seguridad, desarrollar la aplicación móvil android para detección de personas autorizadas y por último construir un prototipo de aplicación móvil android de reconocimiento facial conectada a la placa arduino para proteger el sistema de encendido del vehículo.

El prototipo desarrollado consta de dos partes: la aplicación móvil que se divide en 4 módulos, los cuales tienen una función específica de ayudar en el proceso de detección de personas, administrar dispositivos móviles, administrar dispositivos arduino y ajustes de la aplicación.

El dispositivo electrónico Arduino ha sido construido a partir la plataforma de hardware y software libre Arduino con componentes adicionales para abrir y cerrar conexiones que activen el encendido del vehículo.

La organización del trabajo es la siguiente: en la Sección II se hace un estudio general de los elementos considerados en el desarrollo del prototipo, en la Sección III se explican las características principales del prototipo creado. La Sección IV muestra detalles de la implementación en donde se nombra las tecnologías de desarrollo empleadas. La Sección V muestra

J. Campos, Universidad Nacional de Loja, Loja, Ecuador, jinon.alexix@gmail.com
M. Palma, Universidad Nacional de Loja, Loja, Ecuador, mapalma@unl.edu.ec,
H. Paz, Escuela Politécnica Nacional, Quito, Ecuador, henry.paz@epn.edu.ec

un caso de estudio en el que apoyan las herramientas desarrolladas. Finalmente, se pueden encontrar las conclusiones en el Apartado VI.

II. ESTUDIO GENERAL DE VISIÓN ARTIFICIAL, BLUETOOTH, ARDUINO Y SEGURIDAD.

En este apartado se hace un estudio general de todos los temas analizados para el desarrollo y construcción del prototipo, los cuales se detallan a continuación:

1. Visión Artificial

La visión artificial o visión por computador es la ciencia y la tecnología que permite a las "máquinas" ver, extraer información de las imágenes digitales, resolver alguna tarea o entender la escena que están visionando [1].

Actualmente, las aplicaciones de la visión artificial están muy extendidas y van desde el campo de la industria (contar botellas, comprobar defectos en una cadena de montaje, interpretar un TAC médico...) y el campo de la medicina (recuento y búsqueda de células), hasta los sistemas más complejos, que permiten a los robots orientarse en un entorno desconocido, pasando por el reconocimiento de patrones de la realidad aumentada, entre otras muchas aplicaciones [1].

2. Visión Artificial en dispositivos móviles

Tradicionalmente, la visión por computador es una disciplina científica que ha requerido equipos con gran poder de cómputo. Gracias a los espectaculares avances de la tecnología, los teléfonos móviles de última generación han adquirido capacidades que permiten la ejecución de programas de procesamiento de imagen [2].

Muchas empresas y programadores han empezado a explotar comercialmente aplicaciones de visión en dispositivos móviles.

El interés es también muy grande a nivel académico y científico, como lo atestigua la creciente organización de eventos asociados a congresos científicos internacionales.

Los dispositivos móviles permiten crear aplicaciones de visión por computador muy diversas, algunos ejemplos son [2]:

- Interfaces gestuales
- Reconocimiento óptico de caracteres
- Búsqueda visual
- Realidad aumentada
- Aplicaciones avanzadas de fotografía
- Vigilancia

La librería de funciones más conocida y potente para visión por computador es OpenCV [3]. En la actualidad OpenCV está soportada por las empresas Willow Garage e Itseez y dispone

de más de 2.500 algoritmos. En el momento de escribir estas líneas existe una versión de OpenCV para Android.

3. Bluetooth

Bluetooth es un estándar inalámbrico para interconectar computadoras, dispositivos de comunicaciones y accesorios a través de radios inalámbricos de bajo consumo de energía, corto alcance y económicos [4].

Ahora todas las formas de dispositivos electrónicos para consumidores utilizan Bluetooth, desde los teléfonos móviles y las computadoras portátiles hasta los audífonos, impresoras, teclados, ratones, consolas de videojuegos, relojes, reproductores de música, unidades de navegación, etc.

Los protocolos de Bluetooth permiten a estos dispositivos encontrarse y conectarse entre sí, a lo cual se le conoce como emparejamiento (pairing), además de que pueden transferir datos en forma segura. [4]

La unidad básica de un sistema Bluetooth es una piconet, la cual consta de un nodo maestro y hasta siete nodos esclavos activos a una distancia máxima de 10 metros.

4. Arduino

Arduino es una plataforma electrónica de código abierto basado en hardware y software fácil de usar. Está dirigido a cualquier persona que hace proyectos interactivos. El corazón de Arduino es un microcontrolador. En realidad, el resto de la placa se ocupa de facilitar la alimentación y permitir que se comuniquen con el ordenador al que está conectado [5].

Arduino es un pequeño ordenador en un chip. Tiene todo y más de lo que contenían los primeros ordenadores domésticos. Dispone de un procesador, memoria RAM (memoria de acceso aleatorio) para contener datos, unos cuantos kilobytes de memoria EPROM (ROM programable borrable) o de memoria Flash para contener nuestros programas, y tiene pines de entrada y salida [5].

Las entradas pueden leer señales digitales (¿el interruptor está abierto o cerrado?) y analógicas (¿cuál es la tensión en un pin?). Esto nos permite conectar innumerables tipos de sensores de luz, temperatura, sonido, etc.

Las salidas también pueden ser analógicas o digitales. Así, se puede establecer que un pin esté activado o desactivado (5 V o 0 V) y esto puede encender o apagar los LED directamente, o se puede usar la salida para controlar dispositivos más potentes, como motores.

En la Fig. 1 podemos apreciar una placa de Arduino:



Figura 1. Placa de Arduino UNO.

5. Seguridad

Los problemas de seguridad de se pueden dividir en términos generales en cuatro áreas inter relacionadas: confidencialidad, autenticación, no repudio y control de integridad [4].

- La confidencialidad, consiste en mantener la información fuera del alcance de usuarios no autorizados.
- La autenticación se encarga de determinar con quién se está hablando antes de revelar información delicada o hacer un trato de negocios.
- El no repudio se encarga de las firmas: ¿cómo comprobar que su cliente en realidad hizo un pedido electrónico por 10 millones de utensilios para zurdos a 89 centavos cada uno, cuando después él afirma que el precio era de 69 centavos? O tal vez argumente que él nunca realizó ningún pedido.
- El control de integridad tiene que ver con la forma en que podemos estar seguros de que un mensaje recibido realmente fue el que se envió, y no algo que un adversario malicioso modificó en el camino o ideó por su propia cuenta.

Tomando en cuenta lo anterior, casi toda la seguridad en términos computacionales se basa en principios de criptografía [6].

La Criptografía es una técnica, o conglomerado de técnicas, que tratan sobre la protección, ocultamiento frente a observadores no autorizados de la información [6].

Para el cifrado de datos se ha tomado en cuenta los siguientes algoritmos:

5.1. El algoritmo Rijndael (AES)

En octubre de 2000 el NIST (National Institute for Standards and Technology) anunciaba oficialmente la adopción del algoritmo Rijndael como nuevo Estándar Avanzado de Cifrado (AES) para su empleo en aplicaciones criptográficas no militares, encaminado a proporcionar a la comunidad internacional un nuevo algoritmo de cifrado potente, eficiente, y fácil de implementar, sucesor del DES [6].

AES es un acrónimo formado por los nombres de sus dos autores, los belgas Joan Daemen y Vincent Rijmen. Todo el

proceso de selección, revisión y estudio tanto de este algoritmo como de los restantes candidatos, se ha efectuado de forma pública y abierta, por lo que, prácticamente por primera vez, toda la comunidad criptográfica mundial ha participado en su análisis, lo cual convierte a Rijndael en un algoritmo perfectamente digno de la confianza de todos.

AES es un sistema de cifrado por bloques, diseñado para manejar longitudes de clave y de bloque variables, ambas comprendidas entre los 128 y los 256 bits.

Según sus autores, es altamente improbable que existan claves débiles o semidébiles en AES, debido a la estructura de su diseño, que busca eliminar la simetría en las subclaves. También se ha comprobado que es resistente a criptoanálisis tanto lineal como diferencial [6].

En efecto, el método más eficiente conocido hasta la fecha para recuperar la clave a partir de un par texto cifrado-texto claro es la búsqueda exhaustiva, por lo que podemos considerar este algoritmo como uno de los más seguros en la actualidad. Otro hecho que viene a corroborar la fortaleza de AES es que en junio de 2003 fue aprobado por la NSA para cifrar información clasificada como alto secreto [6].

Ese algoritmo sirve para cifrar y descifrar mensajes de datos siempre y cuando se conozca la clave pública.

Un ejemplo del resultado del algoritmo se detalla a continuación:

Clave pública: "llavemaestra"

Texto a cifrar: "Demostracion de cifrado"

Resultado:

x2Cfr0z9spycz8qLIFMOYhHk9vKXXNGvViMuKHxEsGc=

5.2. MD5

Se trata de uno de los más populares algoritmos de generación de firmas, debido en gran parte a su inclusión en las primeras versiones de PGP. Es el resultado de una serie de mejoras sobre el algoritmo MD4, diseñado por Ron Rivest, procesa los mensajes de entrada en bloques de 512 bits, y produce una salida de 128 bits [6].

El algoritmo toma como entrada un mensaje de longitud arbitraria y produce como salida una "huella digital" de 128 bits de la entrada. Es computacionalmente imposible producir dos mensajes con el mismo hash. El algoritmo MD5 está diseñado para aplicaciones de firma digital, donde un archivo grande debe ser "comprimido" de una manera segura antes de ser cifrada con una clave privada (secreta) en virtud de un criptosistema de clave pública tales como RSA [7].

El algoritmo MD5 para calcular el hash de las claves de los usuarios. En el disco o base de datos se guarda el resultado del MD5 de la clave que se introduce al dar de alta un usuario, y cuando éste quiere entrar en el sistema se compara el hash

MD5 de la clave introducida con el hash que hay guardado en el disco duro o la base de datos, si coinciden la misma clave y el usuario este será autenticado [7].
Un ejemplo del resultado del algoritmo se detalla a continuación:

Texto a cifrar: “Demostracion de cifrado”

Resultado: c3606bb1b231aa56166cc8b46ee3f6c3

III. CARACTERÍSTICAS DE PROTOTIPO DESARROLLADO

El prototipo desarrollado consta de dos partes continuación se describe las características de cada parte que lo compone.

A. APPVCAR

La aplicación móvil se divide en 4 módulos, los cuales tienen una función específica de ayudar en el proceso de detección de personas, administrar dispositivos móviles, administrar dispositivos arduino los cuales se detallarán a continuación:

- 1) Dispositivos Arduino: Este módulo le permite administrar (ver, agregar, eliminar) dispositivos Arduino disponibles.
- 2) Usuarios de aplicación: Le permitirá al administrador gestionar usuarios para el acceso al sistema, en el mismo se podrá crear, modificar, eliminar usuarios.
- 3) Rostros de usuarios: Le permitirá al administrador gestionar (ver, actualizar, eliminar) los rostros de los usuarios.
- 4) Dispositivos móviles: Le permitirá al administrador gestionar (ver, bloquear, eliminar) los dispositivos móviles Android que se agreguen a la aplicación.

B. DISPOSITIVO ARDUINO

El dispositivo electrónico Arduino ha sido construido a partir de la plataforma de hardware y software libre Arduino con componentes adicionales para abrir y cerrar conexiones que activen el encendido del vehículo.

Dentro de los componentes que se ha utilizado tenemos los siguientes:

- Arduino Mega 2560
- Lector de tarjetas SD
- Módulo bluetooth HC-06
- Módulo Relay Arduino
- Resistencias de 200 kΩ
- Led colores Alta luminosidad
- Batería 9V
- Pares de cable macho-hembra

IV. IMPLEMENTACIÓN

Para el desarrollo de cada uno de los componentes se han utilizado diversas herramientas, dando como resultado la aplicación móvil y el dispositivo electrónico que se describen a continuación:

La arquitectura del prototipo se muestra en la Fig. 2



Figura 2. Arquitectura del prototipo.

A. APPVCAR

Para el desarrollo de la aplicación móvil se utilizó el lenguaje de programación JAVA [1] que es el lenguaje con el que trabaja Android [2], el IDE Eclipse Android que es un entorno de desarrollo específico para Android, OpenCV Android [3] que es un conjunto de librerías de visión artificial utilizadas para la detección y reconocimiento de rostros, NDK [4] Android que es una herramienta necesaria para trabajar con OpenCV a nivel de código de bajo nivel.

A continuación se muestra el diagrama (ver Fig. 3) de clases utilizado para el desarrollo de la aplicación móvil.

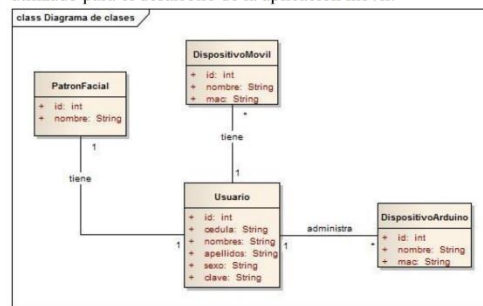


Figura 3. Diagrama de clases.

De la misma manera se muestra el diagrama de Base de datos (ver Fig. 4) donde podemos apreciar como se maneja los datos en la aplicación móvil.

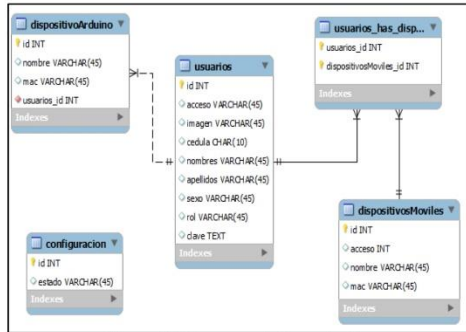


Figura 4. Diagrama de Base de datos.

Además también se muestra la navegabilidad entre las pantallas de la aplicación móvil (ver Fig. 5).

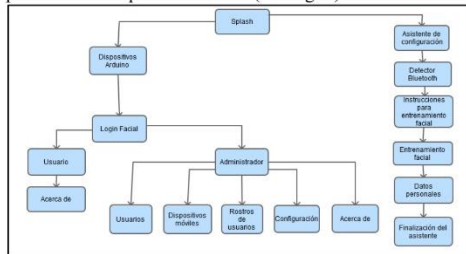


Figura 5. Navegabilidad de pantallas.

B. DISPOSITIVO ARDUINO

Para la construcción del dispositivo electrónico se ha utilizado la tecnología de software y hardware Arduino [5], además también se ha utilizado el software de diseño electrónico Fritzing [11] para realizar los esquemas de conexión y simulación.

A continuación se muestra todas las figuras y tablas de conexión del dispositivo Arduino.

A. CONEXIÓN ARDUINO BLUETOOTH.

En la Fig. 6 podemos apreciar de forma gráfica las conexiones realizadas entre el Arduino y el módulo bluetooth.

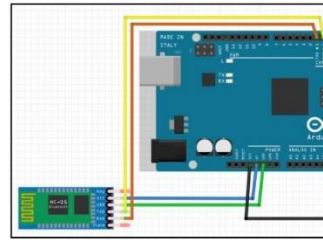


Figura 6. Esquema Conexión Arduino bluetooth.

La tabla 1 contiene la especificación de los pines utilizados en la conexión del Arduino y el módulo bluetooth.

Tabla 1. Conexión de pines arduino bluetooth.

Pin Arduino	Pin bluetooth
5V	VCC
GND	GND
TX	RX
RX	TX

B. CONEXIÓN ARDUINO LECTOR SD.

En la Fig. 7 podemos apreciar de forma gráfica las conexiones realizadas entre el Arduino y el módulo lector de SD.

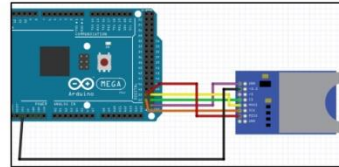


Figura 7. Esquema conexión Arduino Lector SD.

La tabla 2 contiene la especificación de los pines utilizados en la conexión del Arduino y el módulo lector de SD.

Tabla 2. Conexión de pines arduino lector SD.

Pin Arduino	Pin Lector SD
Pin 50	MISO
Pin 51	MOSI
Pin 52	SCK
Pin 53	CS
Pin 3.3V	Pin 3.3V
GND	GND

C. CONEXIÓN ARDUINO MÓDULOS RELAY

En la Fig. 8 podemos apreciar de forma gráfica las conexiones realizadas entre el Arduino y los módulos Relay.

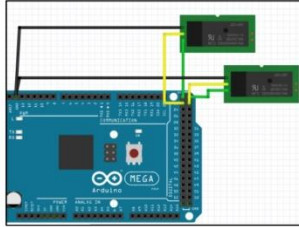


Figura 8. Esquema conexión Arduino Módulos Relay.

La tabla 3 y tabla 4 la especificación de los pines utilizados en la conexión del Arduino y los módulos Relay.

Tabla 3. Conexión de pines arduino modulo relay 1

Pin Arduino	Pin Lector SD
5V	DC+
GND	DC-
Pin 22	IN

Tabla 4. Conexión de pines arduino modulo relay 2

Pin Arduino	Pin Lector SD
5V	DC+
GND	DC-
Pin 23	IN

D. CONEXIÓN ARDUINO LEDS DE ESTADO

En la Fig. 9 podemos apreciar de forma gráfica las conexiones realizadas entre el Arduino y los led que indicarán el estado del dispositivo en funcionamiento.

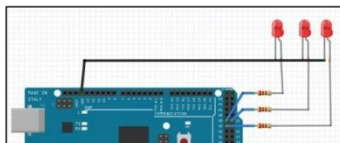


Figura 9. Esquema conexión Arduino leds de estado

La tabla 5, tabla 6 y tabla 7 contienen la especificación de los pines utilizados en la conexión del Arduino y los led.

Tabla 5. Conexión de pines arduino con led actividad.

Pin Arduino	Pin Lector SD
5V	DC+
GND	DC-
Pin 23	IN

Tabla 6. Conexión de pines arduino con led bloqueo eléctrico.

Pin Arduino	Pin Lector SD
5V	DC+
GND	DC-
Pin 23	IN

Tabla 7. Conexión de pines arduino con led bloqueo de seguros.

Pin Arduino	Pin Lector SD
5V	DC+
GND	DC-
Pin 24	IN

E. ESQUEMA FINAL DE CONEXIONES.

En la Fig. 10 podemos apreciar de forma gráfica todas las conexiones realizadas entre el Arduino y todos los componentes mostrados anteriormente.

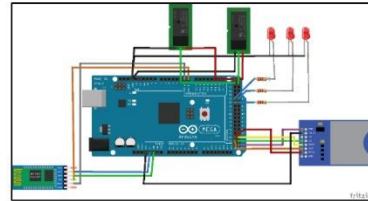


Figura 10. Esquema final de conexiones.

F. VISTA REAL DE PROTOTIPO CONSTRUIDO.

En la Fig. 11 podemos apreciar una vista real del prototipo realizado previo a la puesta en marcha de su funcionamiento.



Figura 11. Vista real superior del prototipo.

V. CASO DE ESTUDIO

En esta Sección se muestra un caso de estudio, donde el prototipo desarrollado se encuentra funcionando:

A. ENTRENAMIENTO FACIAL

En la Fig. 12 podemos observar la pantalla de entrenamiento facial donde presionaremos en el botón Entrenar, Luego en el botón Grabar, con lo cual comenzará a grabar nuestro rostro, una vez que haya terminado de grabar, presionamos de nuevo el botón Entrenando para terminar con el entrenamiento:

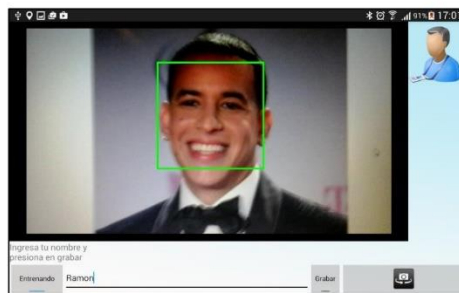


Figura 12. Pantalla Entrenamiento facial.

B. LOGUEO FACIAL

Abrimos de nuevo la aplicación móvil y nos aparecerá la pantalla de dispositivos Arduino registrados, seleccionamos el dispositivo que vamos a utilizar (ver Fig. 13):

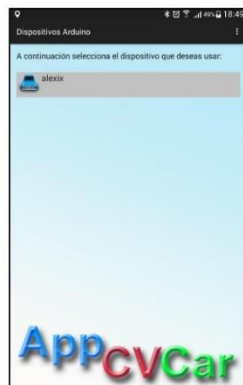


Figura 13. Pantalla Selección de dispositivo Arduino.

Al seleccionar nos aparecerá un cuadro de diálogo presentando dos opciones, presionamos en Desbloquear para continuar(ver Fig 14):

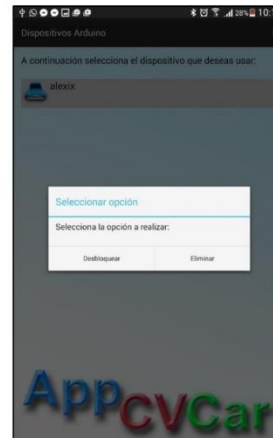


Figura 14. Pantalla Opciones de ingreso

Ahora nos aparece la pantalla de Logueo Facial en donde enfocaremos nuestro rostro y presionaremos en el botón Ingresar para comenzar con el reconocimiento (ver Fig 15):

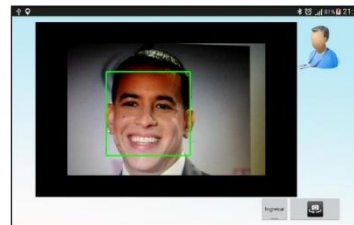


Figura 15. Pantalla Logueo facial

Dependiendo del tipo de usuario nos aparecerá la pantalla de Administración o la pantalla de usuario normal (ver Fig. 16):



Figura 16. Pantalla Administrador

C. ADMINISTRACIÓN DE ROSTROS

En esta sección le permitirá al usuario administrar (modificar, eliminar) los rostros de usuarios registrados (ver Fig. 23).



Figura 17. Pantalla Rostros de usuarios

VI. CONCLUSIONES.

El uso de la plataforma Arduino es una buena opción al momento de trabajar con electrónica y en proyectos de automatización, ya que no se requiere tener un alto nivel de conocimientos en el tema, además de que se convierte en una experiencia interesante al trabajar con este tipo de proyectos.

La Implementación de librerías como OpenCV simplifica de manera significativa el desarrollo de aplicaciones móviles que requieran de funciones que utilizan una cámara como

detección de rostros, reconocimiento de rostros, detectores de movimiento, detectores de objetos etc.

Para el desarrollo de la aplicación móvil se lo ha realizado de forma nativa ya que al trabajar con OpenCV se necesita usar mayores recursos de procesamiento que haciéndolo de forma normal no se obtendría un rendimiento óptimo.

La visión artificial es un campo aún en desarrollo por lo que no se pueden esperar resultados muy exactos, ya que su funcionamiento se limita a la tecnología de cámaras disponibles y de las condiciones de iluminación del ambiente en donde se trabaje.

La metodología de desarrollo de móviles Mobile-D ha resultado muy útil en el presente trabajo de titulación ya que ha permitido un desarrollo ágil considerando que solo se ha contado con un solo miembro de equipo de desarrollo en este caso el tesista.

REFERENCIAS

- [1] J. O. Vera, Introducción al procesamiento digital de imágenes. Un enfoque teórico-práctico utilizando opencv., 2013.
- [2] O. Déniz, J. Salido y G. Bueno, Programación de Apps de Visión Artificial, España: Bubok Publishing S. L., 2013.
- [3] G. Bradsky y A. Kaebler, Learning OpenCV Computer Vision with the OpenCV library, United States: O' Reilly, 2008.
- [4] A. S. Tanenbaum y D. J. Wetherall, Redes de Computadoras, Atalacomulco: Pearson Educación, 2012.
- [5] S. Monk, 30 proyectos con Arduino, Estribor, 2013.
- [6] M. J. Lucena López, Criptografía y Seguridad en computadores, Madrid: Universidad de Jaén, 2010.
- [7] M. L. f. C. S. a. R. D. Security, «The MD5 Message-Digest Algorithm.» [En línea]. Available: <http://www.ietf.org/rfc/rfc1321.txt>. [Último acceso: 2015 11 20].
- [8] J. Gosling, The Java language specification, Addison-Wesley Professional., 2000.
- [9] R. Rogers, Android application development: Programming with the Google SDK, O'Reilly Media Inc., 2009.
- [10] A. developers, «Android NDK,» [En línea]. Available:

<http://developer.android.com/intl/es/tools/sdk/ndk/index.html>. [Último acceso: 25 10 2015].

- [11] F. developer, «Fritzing electronic made easy.» [En línea]. Available: <http://fritzing.org/home/>. [Último acceso: 25 10 2015].



Ecuador, 2015.

Jinsop Campos, egresado de la Carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja periodo 2009-2015. Posee conocimientos de Linux, redes, programación en Java, desarrollo móvil Android, visión artificial, Arduino, sistemas operativos libres y privativos. Áreas de interés: OS Linux y Windows, móviles, Arduino y redes. Provincia de Loja, Ciudad Loja,



Mario Palma, Ingeniero en Sistemas, recibido en la Universidad Nacional de Loja, Magister en Gerencia y Liderazgo Educativo, recibido en la UTPL. Áreas de interés: Aplicaciones móviles y web, Arduino. Provincia de Loja, Ciudad Loja, Ecuador, 2015.



Henry Paz, Ingeniero en Sistemas de la Universidad Nacional de Loja- Ecuador en el 2010, maestro en Ciencias Computacionales de La Universidad Autónoma del estado de Hidalgo-México, en la especialidad de Computación Inteligente en el 2012 Su interés por la investigación actual es el desarrollo de aplicaciones móviles.,

Anexo 2. Glosario de términos

Término	Significado
Aplicación móvil	Aplicación informática diseñada para ser ejecutada en teléfonos inteligentes, tabletas y otros dispositivos móviles.
Aplicación nativa	Aplicación nativa es la que se desarrolla en el lenguaje nativo del propio terminal, dependiendo de la plataforma ya sea Android, IOS o Windows Phone.
Sistema Operativo Móvil	Un sistema operativo es aquel que controla un dispositivo móvil al igual que los PCs que utilizan Windows o Linux.
OpenCV	Es una biblioteca libre de visión artificial multiplataforma utilizada en infinidad de aplicaciones, desde sistemas de seguridad con detección de movimiento, hasta aplicativos de control de procesos donde se requiere reconocimiento de objetos.
Android	Es un sistema operativo móvil basado en el núcleo Linux, muy popular en la actualidad.
OpenCV Android	Es una biblioteca libre de visión artificial diseñada para trabajar en dispositivos Android para aplicaciones que detección de movimiento junto a demás funcionalidades que posee dicha librería.
NDK	Es un conjunto de herramientas que permite implementar en una aplicación partes de código utilizando lenguajes de código nativo, como C y C ++.
Arduino	Es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica.
Entrenamiento facial	Es un proceso en el cual se guardan las características de un rostro para su posterior identificación o reconocimiento.
Reconocimiento facial	Es el proceso en el cual se identifica o reconoce un rostro realizando una comparación con características de rostros guardados previamente.

Dispositivo Arduino	Es un dispositivo electrónico diseñado para proteger el sistema de encendido de un vehículo realizando un corte de corriente en el circuito de dicho sistema.
Bluetooth	Conexión inalámbrica segura diseñada para compartir archivos entre dispositivos móviles dentro de un alcance de no mayor a 10 metros.
Relay	Dispositivo electrónico construido para realizar corte de corriente dentro de un circuito.
Lector SD	Dispositivo electrónico diseñado para leer y escribir datos dentro de una Tarjeta sd.
Cifrado	En criptografía, el cifrado es un procedimiento que utiliza un algoritmo de cifrado con cierta clave (clave de cifrado) transforma un mensaje, sin alterando su estructura lingüística o significado, de tal forma que sea incomprensible o, al menos, difícil de comprender a toda persona que no tenga la clave secreta (clave de descifrado) del algoritmo
MD5	Es uno de los algoritmos de reducción criptográficos diseñados por el profesor Ronald Rivest, fue desarrollado en 1991 como reemplazo del algoritmo MD4.
Fritzing	Es un software de diseño y simulación electrónico libre muy util para construcción de esquemas y circuitos electrónicos
Eclipse ADT	Es un entorno de desarrollo que utiliza el lenguaje de programación Java para la construcción de aplicaciones Android.
IOS	Es un sistema operativo móvil de Apple Inc, originalmente desarrollado para el iPhone, después se ha usado en dispositivos como el iPod touch y el iPad.
Windows Phone	Es un sistema operativo móvil desarrollado por Microsoft, como sucesor de Windows Mobile.

Anexo 3. Licencia del Trabajo de Titulación



Prototipo de detección de personas autorizadas para encender un vehículo aplicando técnicas de visión artificial en dispositivos móviles android by Jinsop Alexix Campos Paredes is licensed under a [Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional License](#).

Anexo 4. Certificado de Resumen



Prof. Joan Morales Abad
DOCENTE DE FINE-TUNED ENGLISH

CERTIFICA:

Que el documento aquí compuesto es fiel traducción del idioma español al idioma inglés de la tesis titulada "PROTOTIPO DE DETECCIÓN DE PERSONAS AUTORIZADAS PARA ENCENDER UN VEHÍCULO APLICANDO TÉCNICAS DE VISIÓN ARTIFICIAL EN DISPOSITIVOS MÓVILES ANDROID" del Sr. Jinsop Alexis Campos Paredes, egresado de la carrera de Ingeniería en Sistemas de la Universidad Nacional de Loja.

Lo certifica en honor a la verdad y autoriza al interesado hacer uso del presente en lo que a sus intereses convenga.

Loja, 11 de Noviembre de 2015




Prof. Joan Morales Abad.
DOCENTE DE FINE-TUNED ENGLISH

Fine-Tuned English Cla. Ltda.

LOJA: Macará entre Miguel Ríos y Rocafuerte * 2578899 * 2563224 * 2574702

www.finetunedenglish.edu.ec

CATAMAYO: Av. 24 de Mayo 08-21 y Juan Montalvo * 2678442

ZAMORA: García Moreno y Pasaje 12 de Febrero * 2608169

