UNIVERSIDAD DE CUENCA

FACULTAD DE INGENIERÍA

CARRERA DE INGENIERÍA ELÉCTRICA

TEMA:

APLICACIONES MULTIMEDIA PARA CONTROL DE UN SISTEMA DOMÓTICO ORIENTADO A PERSONAS CON DISCAPACIDAD

AUTORES:

JOSÉ PATRICIO BARBECHO ASMAL

WILIAN GEOVANNY FERNÁNDEZ NEIRA

Tesis previa a la obtención del Título de Ingeniero Eléctrico

DIRECTOR:

ING. FABIÁN GUSTAVO CABRERA ALBORNOZ

CUENCA - ECUADOR

2012 – 2013

RESUMEN

El proyecto analiza las necesidades básicas de las personas con cierto tipo de discapacidad dentro del hogar. Luego se plantea la creación de tres sistemas de control que permitan a las personas discapacitadas la gestión de cargas eléctricas en una vivienda, estos sistemas permiten al usuario controlar desde un punto fijo y también de forma remota.

El sistema de visión artificial permite controlar cargas a través de señas, para lo cual se cuenta con una caja de adquisición de imágenes, en donde se realiza las instrucciones. Además consta de un sistema de información de obstáculos con notificaciones de voz, para que el individuo tome decisiones de avance o cambio de trayecto.

El sistema para comandos de voz, controla las cargas a través de voz y está diseñado para que interactúe con el usuario, adicionalmente a la aplicación se necesita un micrófono y una computadora.

La aplicación en Android permite el control remoto de las cargas mediante un teléfono celular, siempre y cuando el usuario este dentro del alcance de la red LAN que se ha creado.

Finalmente, se elabora un prototipo para demostrar el funcionamiento y la eficiencia de los sistemas planteados. Así como también, evaluar el impacto que podrá causar en el vivir diario de una persona que tenga cierto tipo de discapacidad.

PALABRAS CLAVES

Discapacidad, domótica, control, visión artificial, comandos de voz, Android, labview, visual basic, interfaz, red, wi-fi.

ABSTRACT

The project analyzes the basic needs of people with certain types of disabilities within the home. Then there is the creation of three control systems that allow disabled people the electrical charges management in a home, these systems allow the user to control from a fixed point and remotely.

The artificial vision system allows the control of loads through signals, for which has an aquisicion box of imagen, where it performs the instructions. Also includes a system information of obstacles with notifications of voice, and the individual decide continue or change the trayectory.

The system for voice commands, controls the loads via voice and is designed to interact with the user, in addition to the application requires a microphone and a computer.

The Android application allows remote control of loads using a cell phone as long as the user is within the reach of the LAN that is created. Finally, a prototype was produced to demonstrate the operation and efficiency of the systems proposed. As well as, evaluating the impact that will have on the daily life of a person who has some kind of disability.

KEYWORDS

Disability, automation, control, vision, voice commands, Android, labview, visual basic, interface, network, wi-fi.

ÌNDICE

INDICE DE GRÁFICOS

ÌNDICE DE TABLAS

Yo, WILIAN GEOVANNY FERNÁNDEZ NEIRA, autor de la tesis "APLICACIONES MULTIMEDIA PARA CONTROL DE UN SISTEMA DOMÓTICO ORIENTADO A PERSONAS CON DISCAPACIDAD", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autor/a.

Cuenca, Septiembre de 2013.

WILIAN GEOVANNY FERNÁNDEZ NEIRA
030238296-5

Yo, WILIAN GEOVANNY FERNÁNDEZ NEIRA, autor de la tesis “APLICACIONES MULTIMEDIA PARA CONTROL DE UN SISTEMA DÓMOTICO ORIENTADO A PERSONAS CON DISCAPACIDAD”, reconozco y acepto el derecho de la Universidad de Cuenca, en base al Art. 5 literal c) de su Reglamento de Propiedad Intelectual, de publicar este trabajo por cualquier medio conocido o por conocer, al ser este requisito para la obtención de mi título de Ingeniero Eléctrico. El uso que la Universidad de Cuenca hiciere de este trabajo, no implicará afección alguna de mis derechos morales o patrimoniales como autor.

Cuenca, Septiembre de 2013.

WILIAN GEOVANNY FERNÁNDEZ NEIRA
030238296-5

Yo, JOSÉ PATRICIO BARBECHO ASMAL, autor de la tesis "APLICACIONES MULTIMEDIA PARA CONTROL DE UN SISTEMA DOMÓTICO ORIENTADO A PERSONAS CON DISCAPACIDAD", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autor/a.

Cuenca, Septiembre de 2013.

JOSÉ PATRICIO BARBECHO ASMAL
010560009-2

Yo, JOSÉ PATRICIO BARBECHO ASMAL, autor de la tesis "APLICACIONES MULTIMEDIA PARA CONTROL DE UN SISTEMA DOMÓTICO ORIENTADO A PERSONAS CON DISCAPACIDAD", reconozco y acepto el derecho de la Universidad de Cuenca, en base al Art. 5 literal c) de su Reglamento de Propiedad Intelectual, de publicar este trabajo por cualquier medio conocido o por conocer, al ser este requisito para la obtención de mi título de Ingeniero Eléctrico. El uso que la Universidad de Cuenca hiciere de este trabajo, no implicará afección alguna de mis derechos morales o patrimoniales como autor.

Cuenca, Septiembre de 2013.

JOSÉ PATRICIO BARBECHO ASMAL
010560009-2

CERTIFICO QUE EL PRESENTE TRABAJO HA

SIDO DESARROLLADO POR LOS SRS:

JOSÉ PATRICIO BARBECHO ASMAL

WILIAN GEOVANNY FERNÁNDEZ NEIRA

_____

Ing. Fabián Gustavo Cabrera Albornoz.
DOCENTE DE LA FACULTAD DE INGENIERIA

DEDICATORIA

El presente proyecto quiero dedicar a dos personas que son muy importantes en mi vida. A mi esposa y mi hija que son el centro de mi inspiración, fortaleza para luchar y seguir adelante. Su apoyo ha sido muy fundamental para culminar esta etapa en mi vida.

Wilian Fernandez.

DEDICATORIA

Este trabajo se lo dedico a mi padres, personas especiales en mi vida, quienes me han brindado apoyo incondicional y han sido motivación para el cumplimiento de mis metas.

Patricio Barbecho.

AGRADECIMIENTO

Gracias a Dios por darme la vida y no abandonarme en los momentos más difíciles.

A mis padres Luis y Mariana por su apoyo incondicional y estar a mi lado enseñándome a luchar por mis sueños. A mis hermanas por su apoyo y fe en mí. A todos mis familiares que de una y otra manera me apoyaron a seguir adelante.

A mis maestros que me llenaron con conocimientos necesarios para superarme cada día, especialmente al Ing. Fabián Cabrera Albornoz por su apoyo como director de tesis.

Wilian Fernandez.

AGRADECIMIENTO

Agradezco a Dios por darme salud, fuerza e insparacion para la realización de este proyecto.

A mi familia quienes fueron apoyo incondicional y de manera especial a mis padres Angel y Maria quienes estuvieron a mi lado apoyandome en los buenos y malos momentos, siempre guiando e incentivando animos para que culmine con éxito mi carrera.

A mis profesores que a lo largo de mi carrera me han llenado de conocimientos, en especial a mi director de tesis Ing. Fabián Cabrera Albornoz, quien con dedicatoria y paciencia supo guiarme para la culminación de este proyecto.

Patricio Barbecho

CAPITULO 1

FUNDAMENTO TEÓRICO

1.1 DISCAPACIDAD EN ECUADOR

1.1.1 Personas con discapacidad en el Ecuador[1]

Actualmente en el Ecuador existe una gran cantidad de personas que padecen cierto tipo de discapacidad, según las estadísticas registradas en el CONADIS (2012) la cantidad de Ecuatorianos con discapacidad es: 41.124 auditiva, 165.700 física, 4.886 lenguaje y 39.455 visual. Por lo que se hace necesario la implementación de nuevos sistemas que faciliten y se incluya a las personas con cierto tipo de discapacidad, para con ello evitar la exclusión de los mismos; en el Ecuador por el alto costo de la tecnología y al no existir equipos para uso universal se hace complejo que los discapacitados con un bajo nivel económico puedan acceder a esta tecnología para su ayuda.

1.1.2 Concepto de discapacidad y clasificación

Concepto de discapacidad.- La Organización Mundial de la Salud define como la restricción o impedimento de la capacidad de realizar una actividad en la forma o dentro del margen que se considera normal para el ser humano. La discapacidad se caracteriza por excesos o insuficiencias en el desempeño de una actividad rutinaria normal, los cuales pueden ser temporales o permanentes, reversibles o surgir como consecuencia directa de la deficiencia o como una respuesta del propio individuo, sobre todo la psicológica, a deficiencias físicas, sensoriales o de otro tipo.

[1] Consejo Nacional de Discapacidades, (2012) CONADIS, Disponible en http://www.conadis.gob.ec/provincias.php

Clasificación de las discapacidades

Discapacidad física: Esta es la clasificación que cuenta con las alteraciones más frecuentes, las cuales son secuelas de poliomielitis, lesión medular (parapléjico o cuadripléjico) y amputaciones.

Discapacidad sensorial: Comprende a las personas con deficiencias visuales, a los sordos y a quienes presentan problemas en la comunicación y el lenguaje.

Discapacidad intelectual: Se caracteriza por una disminución de las funciones mentales superiores (inteligencia, lenguaje, aprendizaje, entre otros), así como de las funciones motoras. Esta discapacidad abarca toda una serie de enfermedades y trastornos, dentro de los cuales se encuentra el retraso mental, el síndrome Down y la parálisis cerebral.

Discapacidad psíquica: Las personas sufren alteraciones neurológicas y trastornos cerebrales.

1.1.3 Calidad de vida de los discapacitados

En los últimos años la calidad de vida de los seres humanos ha tenido un cambio muy significativo, gracias a las Tecnologías de la Información y de la Comunicación, las personas con discapacidad tienen, al alcance de la mano, nuevas formas de comunicación que permiten llevar una vida más autónoma y disfrutar de nuevas fórmulas de ocio, formación y participación, pero al mismo tiempo todo esto ha generado mayores desigualdades y exclusiones para la población que por los altos costos y desconocimiento no pueden tener acceso a los mismos. Este concepto de calidad de vida debe conformarse de los mismos factores y relaciones que son importantes para el resto de las personas.

Las medidas políticas, tecnológicas y sociales a considerar permitirán que las personas con discapacidad se encuentren capacitadas y puedan tener

acceso al desarrollo tecnológico y ocupar el lugar que les corresponden en la toma de decisiones que afecten sus vidas y a través de la tecnología adaptativa y universal poder integrar y dar autonomía a las personas que padezcan cierto tipo de discapacidad.

1.2 AYUDAS TECNICAS

Actualmente existen una gran variedad de ayudas técnicas para abordar diferentes necesidades. Muchas de ellas pueden caer en el ámbito de la tecnología avanzada por dos posibles razones: por ser ayudas técnicas de carácter electrónico o telemático (por ejemplo un sintetizador de voz, una impresora Braille, un teclado de conceptos.); o por ser ayudas técnicas no avanzadas pero necesarias para el acceso o la complementación de equipos o dispositivos avanzados.

Para ello se debe observar a las personas con discapacidad desde sus capacidades y habilidades, su entorno y estilo de vida y así utilizar alguna de las tantas ayudas técnicas, para facilitarles la comunicación y el auto desempeño.

1.2.1 Definición

Las ayudas técnicas se refieren a productos, útiles o equipamientos utilizados para mantener, incrementar o mejorar las capacidades funcionales de las personas con discapacidad[17]. La definición internacional estandarizada, de la ISO9999 para las ayudas técnicas incluye no sólo equipamiento clásico, sino también cualquier herramienta o sistema técnicos para facilitar la movilidad, manipulación, comunicación, control del entorno, y actividades simples o complejas para cualquier aspecto de la vida diaria, la educación o la actividad profesional o social.

1.2.2 Características

Toda ayuda técnica debe reunir las siguientes características:

- **Sencillez:** que permitan un manejo sencillo para poder ser usadas de forma autónoma.

- **Eficacia/utilidad**: respondiendo a las necesidades para las que han sido concebidas y ser utilizadas cuando no existe otro medio razonable de solucionar el problema.
- **Seguridad:** evitando riesgos innecesarios, deben ser fabricadas con materiales resistentes, duraderos, de fácil limpieza, estéticos y de bajo costo.

1.2.3 Clasificación

En función de la utilidad, es decir, del objetivo que persiguen las distintas ayudas técnicas serían:

Ayudas técnica preventivas: aquéllas que previenen deformidades o disminuyen el potencial agresivo y evolutivo de una enfermedad. Habría que tener en cuenta las distintas prevenciones:

1. Primaria: es la que persigue evitar que aparezca la enfermedad.
2. Secundaria: evitar que la enfermedad evolucione hasta la incapacidad.
3. Terciaria: prevención de la dependencia cuando existe incapacidad.

Ayudas técnicas facilitadoras: aumentan las posibilidades funcionales de las personas con discapacidad que las utilizan, distinguiendo las ayudas técnicas de carácter personal o que le afectan directamente y las que se dirigen a la adaptación del hogar o del trabajo (medios técnicos).

Ayudas técnicas compensadoras: aumentan la capacidad de realizar gestos imposibles, bien porque su realización provoque dolor o sea causa de deformidad; bien porque el grado de discapacidad sea tan grande que no pueda efectuarse.

1.2.4 Sistemas alternativos

La tecnología adaptativa puede llegar a reducir el impacto de la discapacidad y satisfacer el derecho de la calidad de vida de las personas con necesidades especiales por tal motivo se ve la necesidad de optar por sistemas y dispositivos que faciliten el vivir diario de la persona discapacitada.

Se puede enumerar varias ayudas para personas con discapacidad visual y/o auditiva entre ellas tenemos:

- **Tecnologías del Habla**: El reconocimiento de voz y la conversión texto-voz.
- **Sistemas multimedia interactivos:** Aquellos que procesan, almacenan y transmiten de forma integrada imágenes, voz, texto y datos con lo cual ofrecen la posibilidad de actuar sobre los contenidos de los mismos, surgiendo así la interactividad.

- **Comunicaciones de avanzada:** La conexión exclusiva a través de las computadoras con la tendencia de incluir la videotelefonía, teléfonos de texto, fax y otros.

- **Sistemas de Acceso:** Interfaces adaptativas que permiten a las personas con discapacidad física o sensorial utilizar una computadora.

- **Sistema de Reconocimiento óptico de caracteres**: Permite a una persona con discapacidad visual reproducir la información desde una computadora utilizando un scanner que lee cualquier texto mediante un programa OCR y los retransmite por medio de un sintetizador de voz o una línea Braille.

- **Pantallas táctiles**: Permiten que personas con dificultades motrices puedan acceder a los movimientos del cursor con la presión de un dedo o mano.

- **Sistemas Alternativos y Aumentativos de comunicación**: Desarrollados para personas que por su discapacidad, no pueden acceder a un código verbal-oral de comunicación

- **Sistemas de Movilidad**: Son aquellos relacionados a la movilidad personal entre estos tenemos brazos o soportes articulados, conmutadores adosados a sillas de ruedas, emuladores de mouse, varillas, micro-robots, etc.

- **Sistemas de Control de Entornos**: permiten la manipulación de dispositivos que ayudan a controlar un entorno.

- **Control Ambiental**: Existe gran número de interfaces que permiten a las personas con discapacidad motora, el poder controlar dispositivos de uso doméstico. Ej. de ello son las llamadas “casas inteligentes", cuyo software facilita: conectar/desconectar timbres, abrir/cerrar puertas, comunicarse por teléfono, control de luces/aire acondicionado/TV u otros dispositivos.

1.2.5 Características de diseño para los sistemas alternativos

El diseño de estos sistemas conjuntamente con sus, entornos, procesos, dispositivos o herramientas deben ser concebidos desde el origen como un diseño universal de tal forma que puedan ser utilizados por el mayor número de personas, considerando que existe una amplia variedad de habilidades humanas y no una habilidad media, sin necesidad de llevar a cabo una adaptación o diseño especializado, simplificando la vida de todas las personas con independencia de su edad, talla o capacidad.

Para conseguir la universalidad del diseño y su fácil implementación se deberá considerar los siguientes aspectos.

Uso equitativo: el diseño debe ser útil y asequible para personas con diversas discapacidades.

Uso flexible: debe ser adaptable a un amplio rango de preferencias y capacidades individuales.

Simple e intuitivo: debe ser fácil de entender, independientemente de la experiencia, conocimiento, nivel cultural o capacidad de concentración.

Información perceptible: debe transmitir de forma eficaz la información necesaria al usuario, con independencia de las condiciones ambientales y de su capacidad sensorial.

Tolerancia a los errores: debe minimizar el peligro y las consecuencias negativas producidas por acciones accidentales o involuntarias.

Bajo esfuerzo físico: debe poder ser utilizado de forma cómoda y eficiente con el mínimo esfuerzo.

1.2.6 Sistemas básicos para una vivienda que alberge discapacitados

En el hogar, en el cual exista personas con cierto tipo de discapacidad se debería adecuar el entorno a las necesidades del usuario con discapacidad de manera temporal o permanente, para ello se debe diseñar herramientas para conseguir, no sólo un ahorro de energía, sino también, la posibilidad de realizar acciones que de otra forma serían imposibles o extremadamente difíciles. Para facilitar el confort y la independencia del discapacitado básicamente se debería tener en cuenta los siguientes factores.

- Encendido y apagado de luces
- Apertura de puertas
- Control de electrodomésticos TV, RADIO
- Control telefónico
- Apertura y cierre de persianas o cortinas
- Control de temperatura
- Seguridad del hogar
- Monitoreo desde el exterior

Para la aplicación de un sistema de control destinado para discapacitados se debe evaluar la posibilidad física de la persona y contar con un sistema de control que permita seleccionar la actuación del sistema para cada tipo de discapacidad.

1.3 DOMÓTICA

1.3.1 Definición

Domótica es el conjunto de automatismos en materia de electricidad, electrónica, robótica, informática y telecomunicaciones, con el objetivo de aumentar el confort, la seguridad, la facilidad de comunicación y entretenimiento del usuario dentro de la vivienda. Pues la domótica busca la integración de todos los aparatos del hogar, de forma que funcionen en perfecta armonía con la mínima intervención por parte del usuario.

El término "domótica" (del latín *domus,* "casa", e informática) tiene varias acepciones, entre ellas la que da el diccionario de la Real Academia, que define a la domótica como "*el conjunto de sistemas que automatizan las diferentes instalaciones de una vivienda.*"[2]

La domótica, en su versión puramente electrónica, es cualesquier solución que permita el control de dispositivos instalados en el hogar. Bajo este concepto permite la gestión de persianas, ventanas, cortinas, electroválvulas, luces, equipos electrónicos cuya actuación sea encendido/apagado (sistemas on/off).

1.3.2 Aplicaciones de la Domótica[3]

A continuación vamos a exponer las aplicaciones más comunes, agrupadas pos áreas:

Confort: Incluye todas las aportaciones que se puedan llevar a cabo para mejorar el confort en una vivienda. Como pueden ser:

- Iluminación:

- Apagado general de todas las luces de la vivienda.

[2] Consejería de la economía e innovación tecnológica. ***"La Domotica como Solucion del Futuro".*** Madrid ,2007.

[3] http://www.bhtavanza.com/es/ingenieria/ingenieriaihd/57-proyectos-domotica

- Automatización del apagado/ encendido en cada punto de luz.
- Regulación de la iluminación según el nivel de luminosidad ambiente.
- Automatización de todos los distintos sistemas/ instalaciones / equipos dotándolos de control eficiente y de fácil manejo (climatización, electrodomésticos, hilo musical, video multimedia, etc.).

- Control de la temperatura de la vivienda por zonas.
- Automatización de persianas y toldos, para que interactúen según las condiciones meteorológicas o la luminosidad.
- Integración del portero al teléfono, o del video portero al televisor.
- Control vía Internet del sistema domótico de la vivienda.
- Gestión Multimedia y del ocio electrónico.
- Generación de escenas y programaciones de forma sencilla para el usuario.
- Sistema de riego inteligente.

Ahorro energético: El ahorro energético no es algo tangible, sino un concepto al que se puede llegar de muchas maneras. En muchos casos no es necesario sustituir los aparatos o sistemas del hogar por otros que consuman menos sino una gestión eficiente de los mismos. Para conseguir esta eficiencia, los sistemas domóticos nos pueden ser de gran ayuda:

- Climatización y consumo energético: programación de los aparatos de clima y zonificación.

- Gestión eléctrica:

- Racionalización de cargas eléctricas: desconexión de equipos de uso no prioritario en función del consumo eléctrico en un momento dado.

- Gestión de tarifas, derivando el funcionamiento de algunos aparatos a horas de tarifa reducida.

- Monitorización de consumos.
    - Uso de energías renovables.
    - Apagado general cuando salimos de la vivienda.
    - Tratamiento y depuración de aguas pluviales.

Seguridad: Consiste en una red de seguridad que se comunica con el sistema domótico encargada de alertar tanto de alarmas técnicas como de alarmas de intrusión:

- Simulación de presencia.
- Alarmas técnicas: incendio, fugas de gas, escapes de agua, concentración de monóxido en garajes. etc.
- Alerta médica (Teleasistencia).
- Cerramiento de persianas.
- Acceso local o remoto a Cámaras IP.
- Avisos al móvil.
- Barreras perimetrales.
- Conexión a CRA (central receptora de alarmas).
- Control de accesos.

Comunicaciones: Son los sistemas o infraestructuras de comunicaciones que posee el hogar.

- Ubicuidad en el control de la vivienda tanto en el exterior como en el interior, control remoto desde Internet, PC, mandos inalámbricos, teléfono móvil, PDA, etc.
- Redes multimedia del Hogar.
- Avisos de alarma.
- Interfaces de voz.

Accesibilidad: Gracias a la integración de tecnología en las viviendas las personas con discapacidad pueden acceder a estas tecnologías sin temor a un obstáculo del tipo tecnológico o arquitectónico.

Figura 1. Esquema de una instalación domótica.

1.3.3 Tipos de Arquitecturas[4]

Como en cualquier sistema de control, la arquitectura especifica el modo en que los diferentes elementos de control del sistema se van a ubicar. Los que se suele denominar arquitectura de control de la red. Pueden ser de varios tipos:

- *Arquitectura centralizada*: Es aquella en la que los elementos a controlar y supervisar (sensores, luces, válvulas, etc.) han de cablearse hasta el sistema de control de la vivienda (PC o similar). Es decir, tiene un único nodo que recibe toda la información de las entradas, que las procesa y envía a las salidas las órdenes de acción correspondiente.

[4] *Curso de domótica e inmótica*, JUAN ANTONIO, 2009

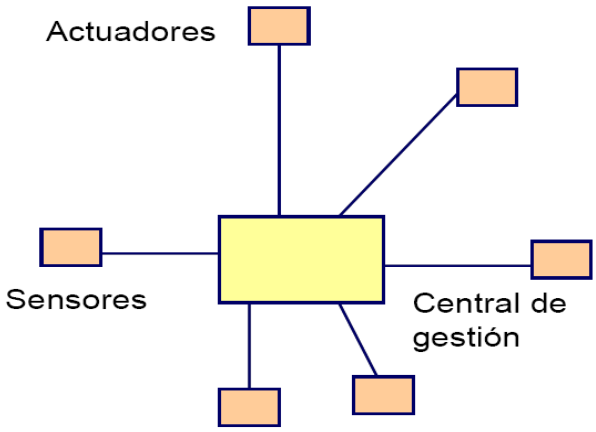


Figura 2. Arquitectura centralizada.

- *Arquitectura descentralizada*: En la arquitectura descentralizada todos los elementos de la red actúan de forma independientemente unos de otros. Comparten la misma línea de comunicación y cada uno de ellos dispone de funciones de control y de mando.

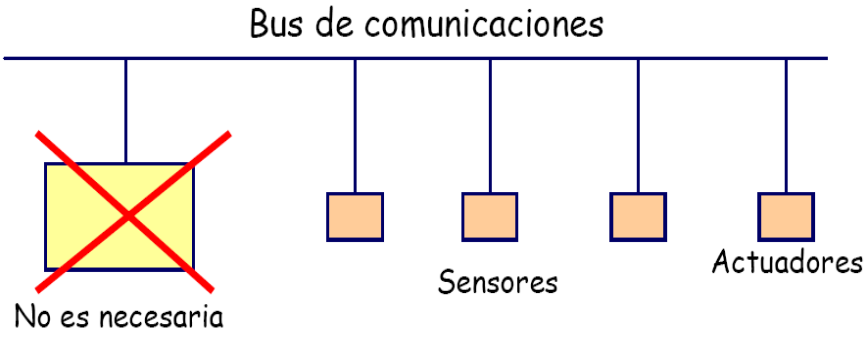


Figura 3. Arquitectura descentralizada.

- *Arquitectura distribuida (híbrida):* Es la combinación de las arquitecturas centralizadas y descentralizadas. En la cual elemento de control se sitúa próximo al elemento a controlar y cada unidad de control esta comunicada mediante un bus de datos.

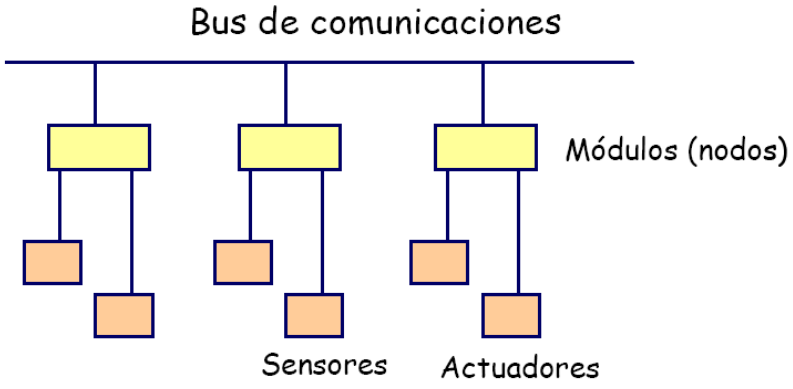


Figura 4. Arquitectura distribuida (hibrida).

1.3.4 Topología del Sistema[5]

Otro aspecto que caracteriza a un sistema es su topología, es decir, la organización física y lógica de los "nodos" los cuales pueden ser de varios tipos:

- **Estrella:** los dispositivos de entrada (sensores) y los de salida (actuadores) van cableados hasta la unidad de control donde se procesa los datos del conjunto. Es decir, posee un control centralizado.

- **Anillo:** los nodos se conectan en un bucle cerrado y los datos se transmiten de nodo en nodo alrededor del bucle, siempre en la misma dirección.

- **Bus:** todos los elementos del sistema (sensores, actuadores y nodos) están ligados sobre una línea que describe el conjunto o parte de una red.

- **Árbol:** es una topología que combina a los demás de esta manera se consigue un mejor control del sistema.

[5] *Curso de domótica e inmótica*, JUAN ANTONIO, 2009

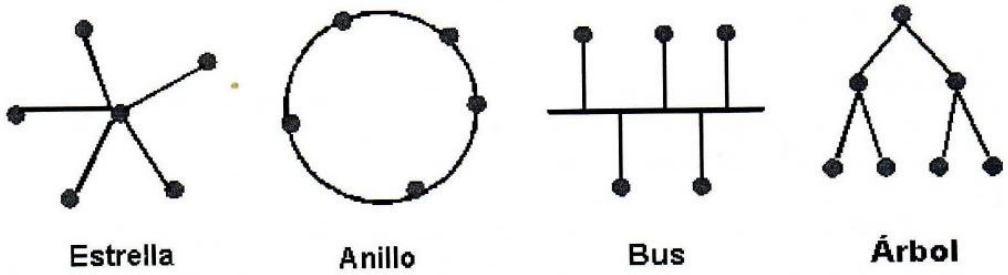


Figura 5. Topologías de un sistema domótico.[6]

1.3.5 Medios de Transmisión

Para que los elementos de un sistema domótico se comuniquen e intercambien información entre sí, se usan medios de transmisión que pueden ser:

- *Sistema de corrientes portadoras*: usa señales que se acoplan y transmiten por la instalación eléctrica de baja tensión.
- *Sistema de transmisión por cables:* usa señales transmitidas por cables específicos para dicha función, tales como cables de par trenzado, paralelo, coaxial, fibra óptica, etc.
- *Sistema de transmisión radiada:* usa señales radiadas tales como ondas infrarrojas, radiofrecuencias, ultrasonido, wireless, bluetooth, etc.

1.3.3 Protocolos de Comunicación[7]

Un sistema domótico se caracteriza por el protocolo de comunicaciones que utiliza, que no es otra cosa que el idioma que los diferentes elementos de control del sistema deben utilizar para entenderse unos con otros y que les permite intercambiar su información de una manera coherente.

Dentro de los protocolos existentes, se puede realizar una primera clasificación atendiendo a su estandarización:

[7] Pablo Moisés Acosta Luque, Juan José Rivadeneira Astudillo. *"Diseño e Implemetacion de un Sistema IVR para Telecontrol Domotico por medio de un Telefono Movil"*. Salgolquí- Ecuador, 2006

- *Protocolos propietarios*. Son aquellos que, desarrollados por una empresa, solo son capaces de comunicarse entre sí.

- *Protocolos estándar*. Los protocolos estándar son los que de alguna manera son utilizados ampliamente por diferentes empresas y éstas fabrican productos que son compatibles entre sí, como son el X-10, BACnet, BatiBus, CEBus, EHS, EIB, HBS, HES, Konnex, dupline y LonWorks.

1.3.4 Dispositivos para el Control y la Automatización

Los dispositivos que se deben instalar en una vivienda para lograr su automatización y control son, básicamente, los siguientes:

- **La pasarela residencial** es el dispositivo que interconecta a los diferentes dispositivos destinados a la automatización de una vivienda, haciendo de interfaz común de todos ellos hacia las redes externas. Permite también el control local y remoto de todos los dispositivos de la vivienda.

- **El sistema (o sistemas) de control centralizado** es el dispositivo encargado de controlar a los dispositivos destinados para la autorización, según los parámetros de actuación establecidos por los usuarios.

- **Los sensores** estos convierten las señales físicas en eléctricas y están encargados de recoger la información de los diferentes parámetros que controla el sistema de control centralizado (la luminosidad, temperatura, fugas de agua, gas, presencia de un intruso, etc.) y enviársela a la pasarela residencial para que ejecuta las tareas programadas. Existen una gran variedad de sensores (de luz, movimiento, gas, agua, etc.) que se pueden usar según la necesidad.

- **Los actuadores** son dispositivos usados por el sistema de control centralizado, para modificar el estado de ciertos equipos o instalaciones. Hay de diversos tipos (contactores de carril DIN, electroválvulas, sirenas, etc.) que se pueden usar según la necesidad.

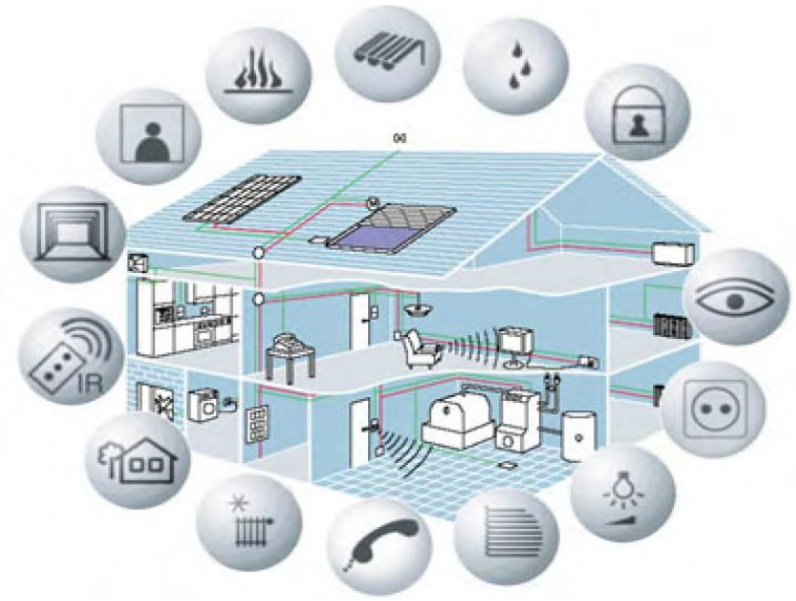


Figura 6. Automatización y control de un sistema domótico.

1.4 VISIÓN ARTIFICIAL

1.4.1 Definición

La visión artificial es un subcampo de la inteligencia artificial que trata de simular el proceso de sentido de la visión de los seres vivos a través de un computador que mediante el uso técnicas adecuadas, permite la obtención, procesamiento y análisis de cualquier tipo de información especial obtenida a través de imágenes digitales.

También es conocida como visión por computador (del inglés computer vision) o visión técnica. Cuyo propósito es programar a un computador para

que “entienda” una escena o las características de una imagen con el propósito de realizar aplicaciones para fin determinado.[8]

Un sistema de visión artificial está compuesto básicamente por:

- **Hardware.-** Dentro del cual consta un subsistema de iluminación, el cual está formado por componentes que permiten condiciones de iluminación uniformes e independientes del entrono y un subsistema de adquisición que está encargado de recoger características del objeto en estudio a través de una imagen digital.

- **Software.-** Subsistema de procesamiento de imágenes, cuya función es realizar el análisis de la imagen mediante un conjunto de algoritmos transformaciones etc. Con la finalidad de extraer la información de la imágenes capturadas.

En la figura 7 se puede observar los componentes que intervienen en un sistema de visión artificial.

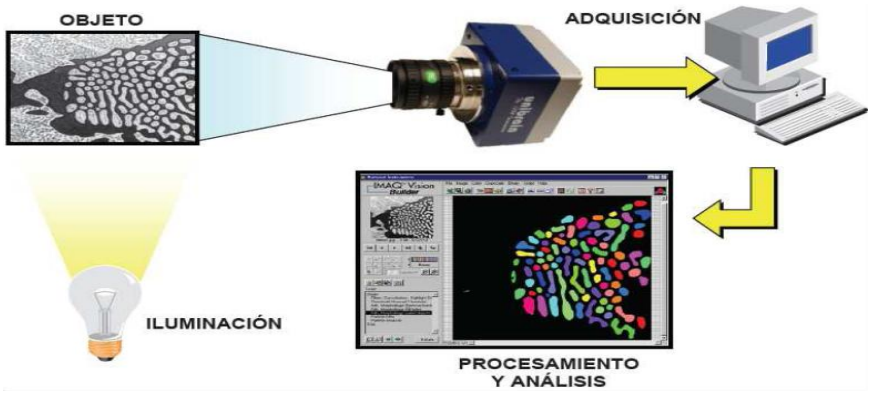


Figura 7. Componentes de un sistema de visión artificial.

[8] **“***Sistema de Sensacion, Segmentacion, Reconocimiento y Clasificacion de Objetos”,* Prof. M.Sc. Kryscia Daviana Ramírez Benavides.

1.4.2 Definición de Imagen

La imagen es una pintura, una fotografía o alguna forma de representación visual de un objeto o escena, cuya proyección de mundo de tres dimensiones es convertida a dos dimensiones, para luego ser digitalizada.

Por su parte la imagen digital es una matriz de dos dimensiones de números reales representados por un número finito de bits. La imagen al digitalizarse se compondrá de pequeños elementos cuadriculados denominados ***pixeles***, a las cuales se les asigna un valor numérico que lleva la información de la imagen.

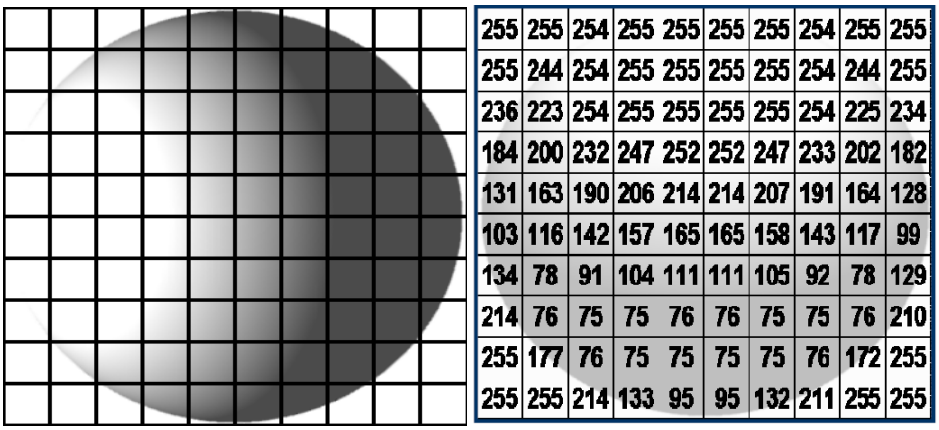


Figura 8. Matriz de una imagen digital.

Mientras mayor es el tamaño de los pixeles mayor es la perdida de información de la imagen, esta pérdida de información se le denomina "*aliasing".*

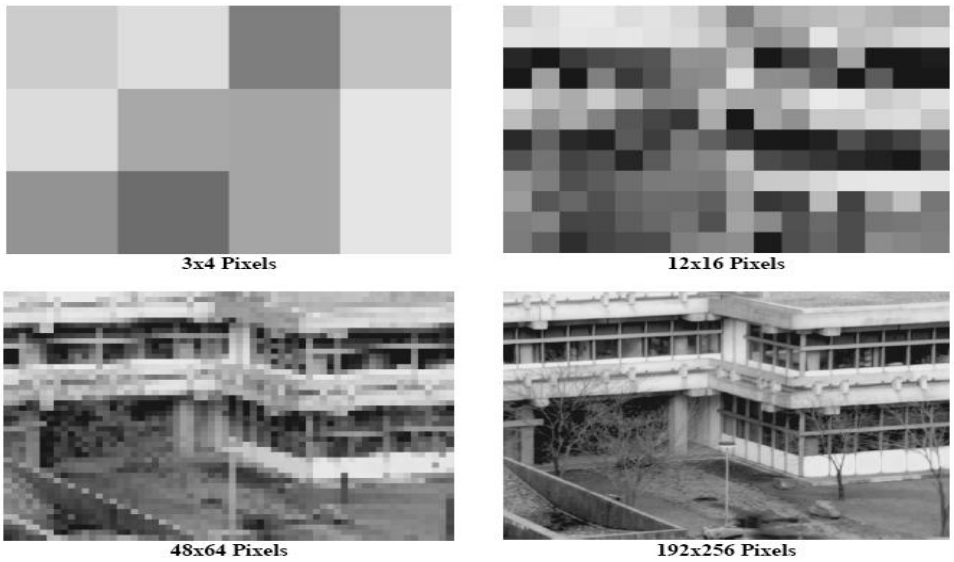


Figura 9. Imágenes con diferente número de pixeles.

1.4.3 Etapas del Procesamiento de Imágenes

El campo del Procesamiento Digital de Imágenes se refiere al estudio de técnicas que permitan de alguna manera mejorar una imagen, de modo que pueda ser utilizada en etapas posteriores de procesos de visión.

Obviamente, para poder extraer información de las imágenes, estas deben tener una buena calidad, y es justamente aquí donde encaja el Procesamiento Digital de Imágenes para mejorar dichas imágenes. Desde este punto de vista, el procesamiento de imágenes involucra tareas como eliminación de ruido, mejoramiento del contraste, segmentación, representación, interpretación, etc.

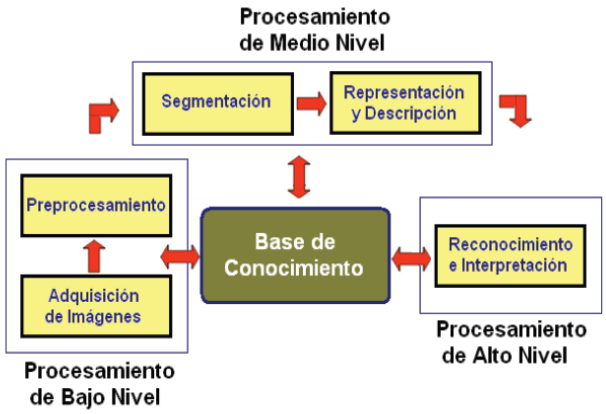


Figura 10. Etapas del procesamiento de imágenes.

1.4.3.1 Captación o Adquisición de Imágenes

Es el proceso mediante el cual se obtiene una imagen para lo cual se emplean cámaras y tarjeta s de adquisición de imágenes. Al igual que el ojo humano las cámaras formadas por un lente convergente proyectan la imagen sobre una superficie sensible a la luz denominada sensor de imagen la cual permite transferirla a un sistema electrónico.

CÁMARAS

Las cámaras utilizadas en visión artificial requieren de una serie de características que permitan el control del disparo de para capturar piezas que pasan por delante de ella en la posición requerida. Son más sofisticadas que las cámaras convencionales, ya que tienen que poder realizar un control completo de: tiempos señales, velocidad de obturación, sensibilidad, etc.

Se clasifican por:

- La tecnología del elemento sensor.
  - Cámaras de tubo. Se basan en la utilización de un material fotosensible que capta la imagen, siendo leída por un haz de electrones.
  - Cámaras de estado sólido CCD (Charge – Coupled – Device). Se basan en materiales semiconductores fotosensibles para cuya lectura no es necesario un barrido electrónico (más pequeñas que las de tubo).

- La disposición física.
  - Cámaras lineales. Se basan en un sensor CCD lineal.
  - Cámaras matriciales. Se basan en un sensor CCD matricial, lo que permite el análisis de imágenes bidimensionales.

Hay una cámara específica para cada aplicación, color, monocromo, alta definición, alta sensibilidad, alta velocidad, infrarrojas, etc. Pasamos a comentar en forma breve el funcionamiento de las más utilizadas.

- **Cámaras lineales.-** Construyen la imagen línea a línea realizando un barrido del objeto junto con un desplazamiento longitudinal del mismo. Las cámaras lineales utilizan sensores que tienen entre los 512 y 8192 pixeles, con una longitud lo más corta posible y gran calidad de imagen.

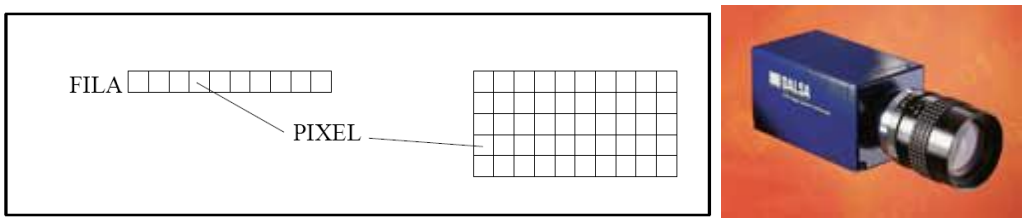


Figura 11. Cámara lineal.

- **Cámaras matriciales.-** El sensor cubre un área que está formada por una matriz de pixeles. Los sensores de las cámaras modernas son todos de tecnología CCD formados por miles de diodos fotosensibles posicionados de forma muy precisa en la matriz.

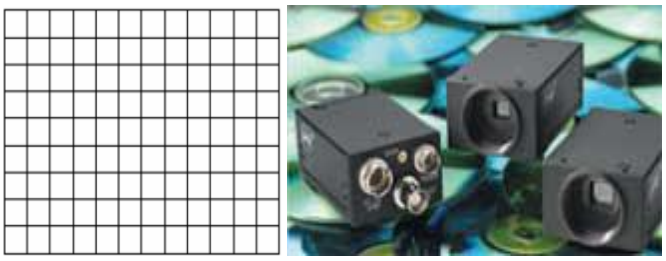


Figura 12. Cámara matricial.

1.4.3.2 Pre-Procesamiento

Es el conjunto de métodos y técnicas que permiten quitar o reducir características no deseadas en la imagen como el ruido. Así como también el realce o mejoramiento de algunas características importantes en las imágenes originales para facilitar el proceso de segmentación.

Cuando se adquiere una imagen esta no puede ser usada directamente en el sistema de visión artificial debido a deficiencias en la iluminación, intensidad de ruido, etc. Por lo cual se hace necesario el pre-procesamiento

para corregir estos errores y además usar transformaciones para resaltar ciertas características que se deseen extraer de la imagen.

Algunas etapas de pre-procesamiento más habituales son:

- **Transformación a escala de grises.-**Se utiliza para filtrar ruido**,** suavizar la intensidad de pixel o extraer características en nivel de gris.

Cada píxel de una imagen en escala de grises puede ser uno de los 256 valores distintos de gris en un sistema de 8 bits, del negro (cero) al blanco (255). Este tipo de datos muestra suaves cambios de tono utilizando tonos intermedios de gris.

Figura 13. Transformación a escala de grises.

- **Binarización de la imagen.-** Una imagen binaria es una imagen en la cual cada píxel puede tener solo uno de dos valores posibles 1 o 0. Como es lógico suponer una imagen en esas condiciones es mucho

más fácil encontrar y distinguir sus características estructurales. La forma más común de generar imágenes binarias es mediante la utilización del valor umbral de una imagen a escala de grises; es decir se elige un valor límite (o bien un intervalo) a partir del cual todos los valores de intensidades mayores serán codificados como 1 mientras que los que estén por debajo serán codificados a cero. Además por medio de la binarización se consigue eliminar el efecto del brillo en la imagen y prepararla para la etapa de procesamiento y extracción de características.

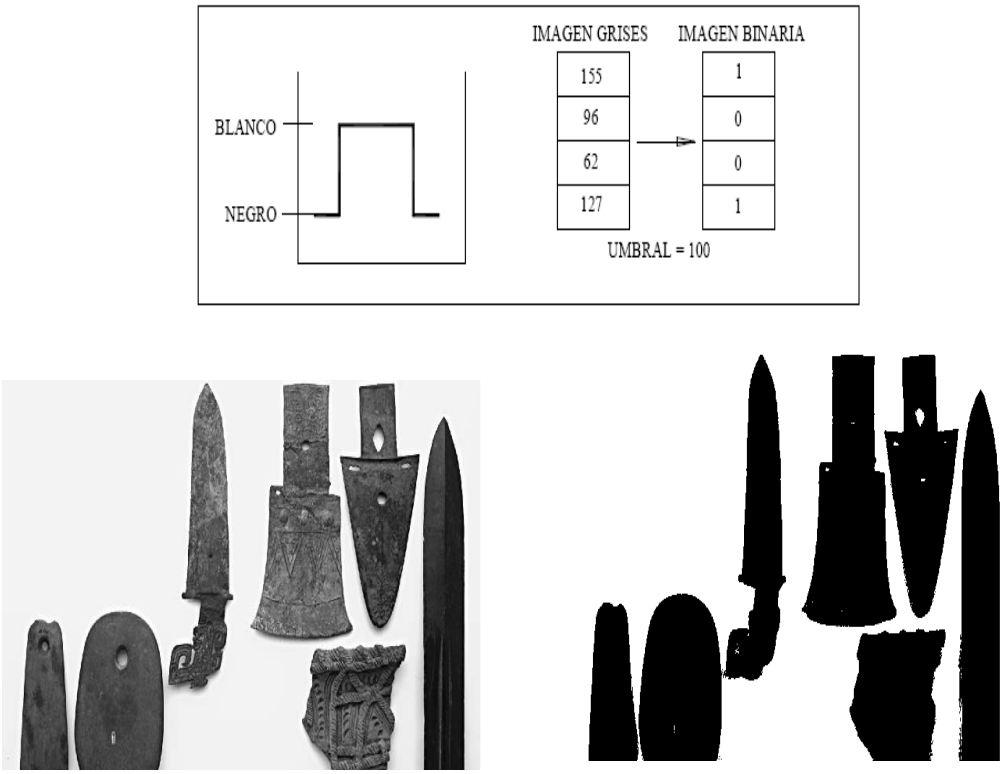


Figura 14. Binarización de una imagen con un umbral de 204.

- **Transformaciones Geométricas.-** No alteran la información de la imagen sino su objetivo es transformar la imagen para apreciarla desde otro punto de vista por ejemplo: rotar, trasladar, zoom, etc.

Figura 15. Transformación geométrica.

- **Histograma de la imagen.-**El histograma es una herramienta en el cual una imagen es representada por la frecuencia de aparición de los distintos niveles de gris en la imagen y proporciona los niveles de gris pero ignorando sus coordenadas. Con el uso del histograma se reduce notablemente la carga computacional.

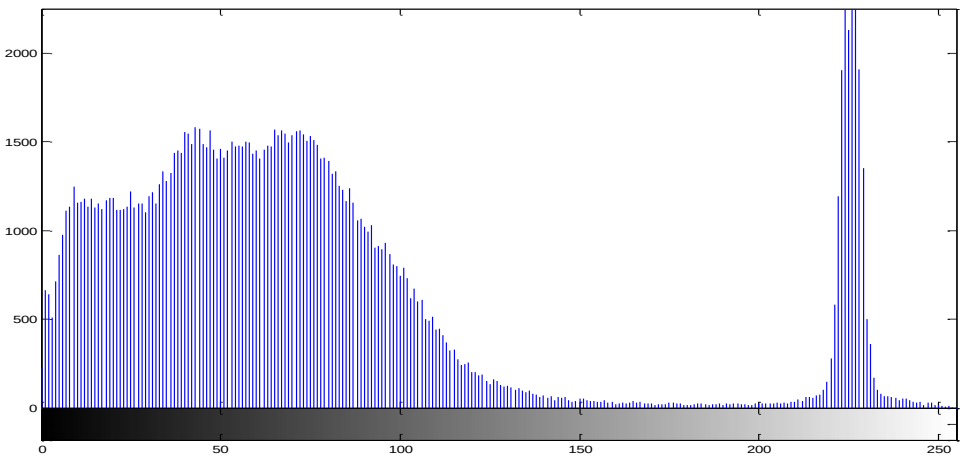


Figura 16. Histograma de la imagen.

- **Variación del contraste.-** La variación o realce del contrate permite mejorar algunas características visuales y eliminar de cierto modo la falta de iluminación uniforme en la imagen. Una herramienta muy útil para este fin es el histograma.

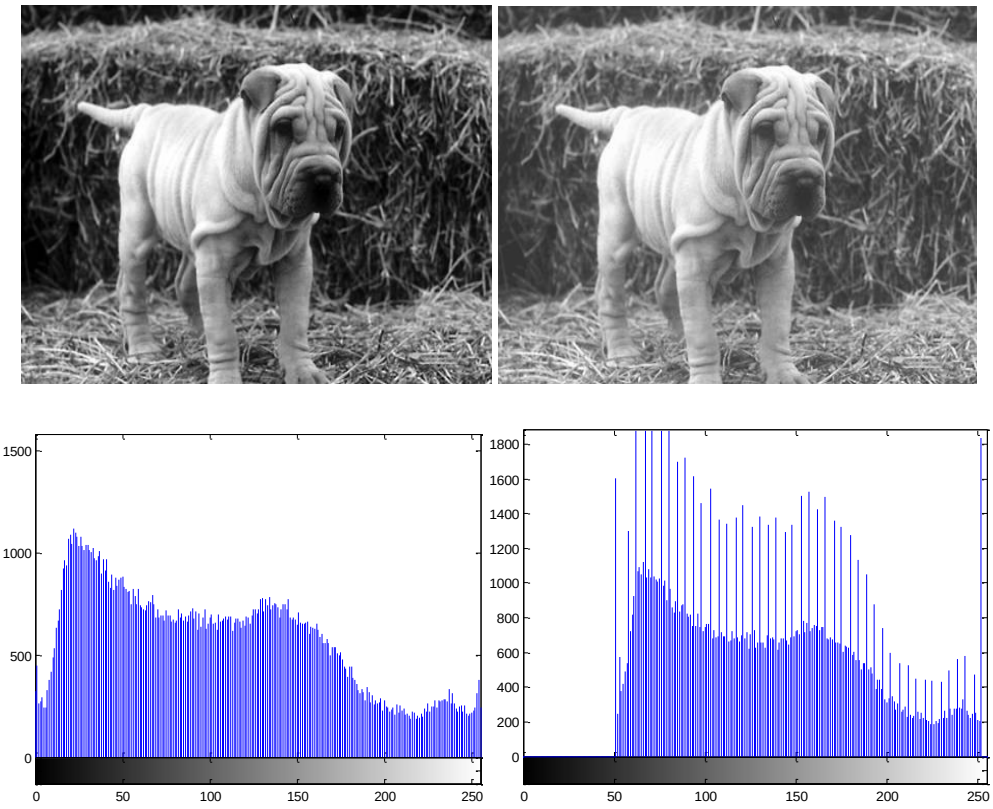


Figura 17. Variación de contraste.

- **Suavizado o eliminación de ruido.-** En el proceso de formación de la imagen se genera cierta información que no es de nuestro interés (ruido) que es necesario descartar para lo cual existen ciertas técnicas denominadas filtros entre los cuales tenemos:

  - Filtros pasa bajos.
  - Filtros gaussianos.
  - Filtros basados en la mediana.
  - Filtros homomórficos. Etc

- **Detección de bordes.-** Son técnicas que se usan antes de la segmentación o para la detección de bordes por su geometría. Un

borde se define como una zona donde existe una fuerte variación del nivel intensidad en los pixeles adyacentes.

Entre estas técnicas tenemos las siguientes:

- Técnica basada en el operador gradiente (Roberts, Prewitt, Sobel, Frei-Chen)
- Técnica basada en el operador lapaciano.
- Técnica basada en el operador Canddy

1.4.3.3 Segmentación

La segmentación es el proceso de dividir una imagen digital en varias partes (grupos de píxeles) u objetos. El objetivo es simplificar y/o cambiar la representación de una imagen en otra más significativa y más fácil de analizar.

Los algoritmos de segmentación se basan en los siguientes principios:

- **Discontinuidades del nivel de gris.** Consisten en segmentar la imagen a partir de los cambios grandes en los niveles de gris entre los píxeles. Las técnicas que utilizan las discontinuidades como base son la detección de líneas, de bordes, de puntos aislados, etc.

- **Similitud de niveles de gris.** Es lo contrario al método anterior, las divisiones de la imagen se hacen agrupando los píxeles que tienen unas características similares. Algunas técnicas que usan esto son la umbralización, el crecimiento de regiones, etc.

- **Conectividad.-** Establecen una conexión o camino entre agrupaciones de pixeles. Cabe mencionar que la conectividad es la forma como los pixeles de una imagen están interconectados, así que

cada pixel está rodeado por pixeles vecinos teniendo los siguientes casos:

- 4Vecinos N 4: un pixel p(x,y) tiene 4 vecinos, 2 horizontales y 2 verticales cuyas coordenadas son:(x+1,y),(x-1,y), (x,y+1),(x,y-1).

- 4 vecinos diagonales N D: un pixel p(x,y) tiene 4 vecinos en diagonal cuyas coordenadas son: :(x+1,y+1),(x+1,y-1), (x-1,y+1),(x-1,y-1).

- 8 vecinos N8: un pixel p(x,y) tiene 4 vecinos, 2 horizontales y 2 verticales, mas 4 vecinos en diagonal cuyas coordenadas son: x+1,y),(x-1,y),(x,y+1),(x,y-1),(x+1,y+1),(x+1,y-1),(x-1,y+1),(x-1,y-1).

En la siguiente figura se puede apreciar las vecindades de un pixel.

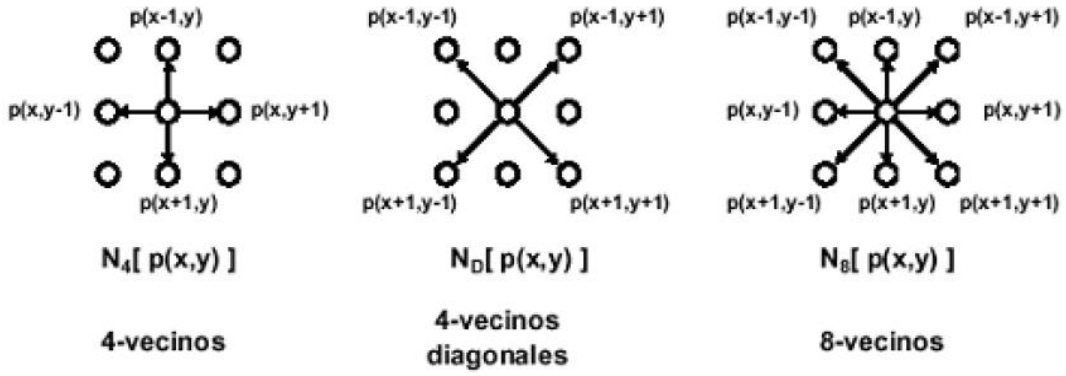


Figura 18. Vecindad de un pixel

Existen varios algoritmos que se usan para la segmentación como son:

- Métodos de agrupamiento (Clustering) – Algoritmo de K-means.
- Métodos basados en el histograma.
- Detección de bordes.
- Métodos de crecimiento de regiones.

- Método del conjunto de nivel.
- Métodos de particionamiento gráfico.
- Transformación divisoria (watershed).
- Método del valor umbral (umbralización).
- Segmentación basada en modelos.
- Segmentación basada en morfología
- Segmentación multi-escala.
- Segmentación semi-automática.
- Redes neuronales de segmentación.

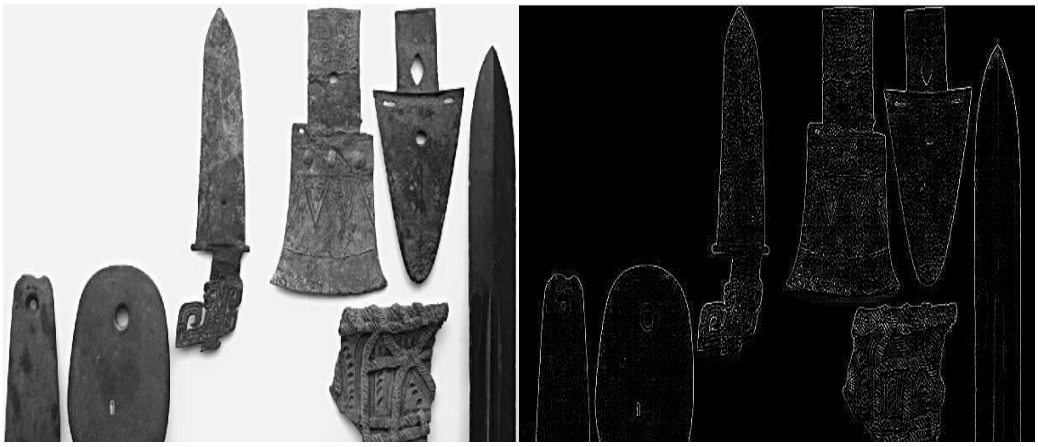
A continuación pasamos a comentar en de forma breve los más comunes.

Segmentación por detección de bordes

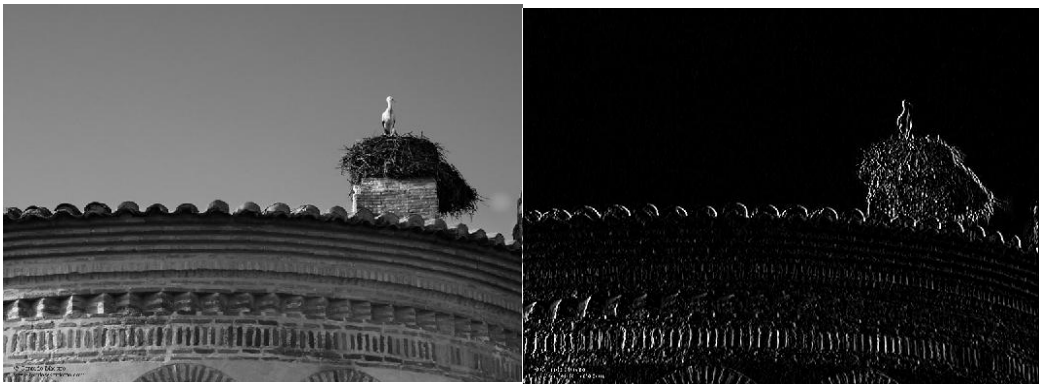
Este tipo de segmentación consiste en agrupar pixeles determinados como los que forman los bordes usando la característica de conectividad. Para que un pixel de borde se defina como un pixel de frontera se necesita que otros pixeles tengan similar dirección y modulo del gradiente.

La detección de bordes es particularmente importante porque proporciona información de la imagen que se destina a otras tareas del procesamiento de imágenes como reconocimiento e interpretación. Este procedimiento es el más común para la detección de discontinuidades cuyo objetivo es resaltar los bordes de los objetos presentes en la imagen.

En la siguiente figura se muestran ejemplos por detección de bordes.



(a)



(b)

Figura 19. Detección de bordes: a) Laplaciano, b) Prewitt.

Umbralización

La umbralización es una técnica de segmentación ampliamente utilizada en las aplicaciones industriales. Se emplea cuando en una imagen hay una clara diferencia entre los objetos a extraer respecto del fondo de la escena. Los principios que rigen son la similitud entre los píxeles pertenecientes a un objeto y sus diferencias respecto al resto. Por tanto, la escena debe caracterizarse por un fondo uniforme y por objetos parecidos.

Al aplicar un umbral, la imagen de niveles de grises quedará binarizada. Conjuntamente con esta técnica se emplea la segmentación por histograma dado a que en este tipo de imágenes comúnmente están conformadas de

dos picos y un valle. Para estos casos el valor umbral estará justo en el valle.

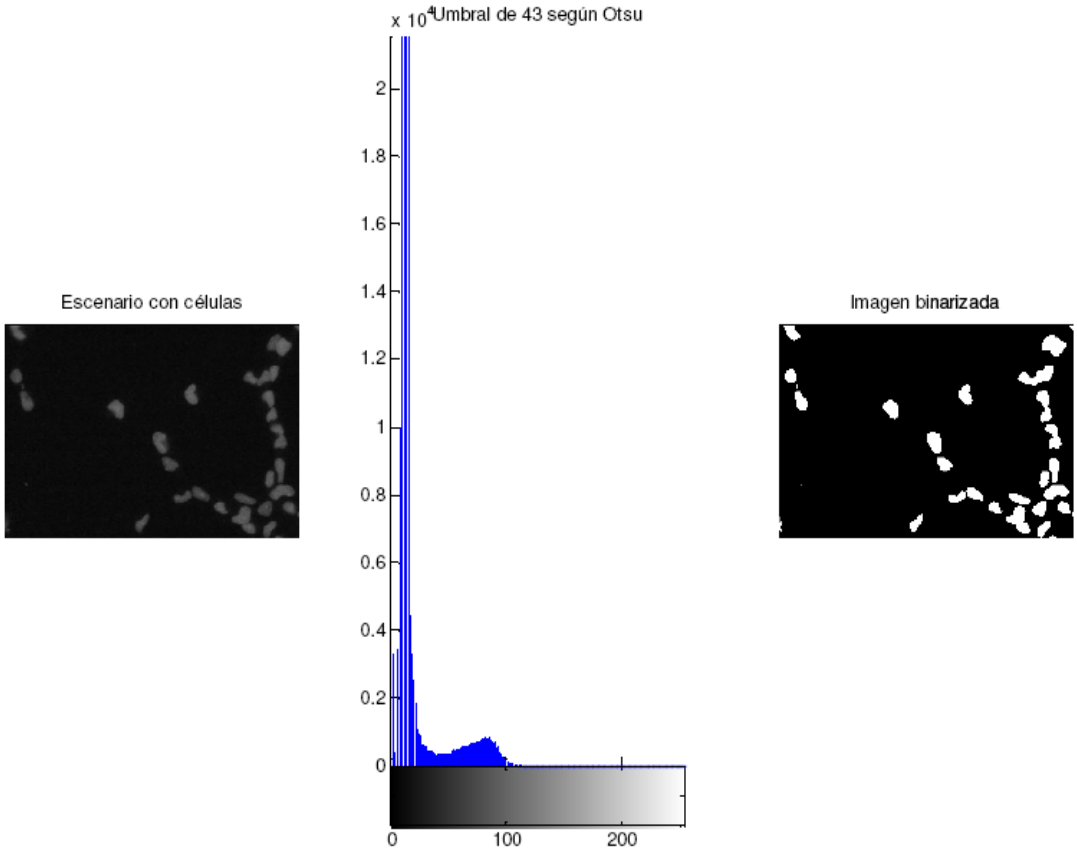


Figura 20. Umbralización mediante histograma.

Al utilizar este método de umbralización se presentan ciertas limitaciones.

- Difícil identificación de los mínimos en el histograma.
- No se pueden distinguir regiones diferentes con similar nivel de gris.
- Se aplica cuando solo hay pocas regiones
- No considera la conectividad entre pixeles.

Segmentación basada en morfología

Se basa en el realce de la geometría y forma de los objetos presentes en la imagen. Para esto se usa una serie de transformaciones morfológicas.

El objetivo de estas transformaciones morfológicas es la extracción de estructuras geométricas del conjunto sobre el cual opera, para lo cual se usa un elemento de forma conocida el que se denomina elemento estructurante. El tamaño y la forma del elemento estructurante se elige en función de las formas que se desea extraer. Muchas veces el proceso morfológico es usado después de la segmentación por bordes.

- **Dilatación**

Una operación de dilatación aumenta el tamaño de los objetos mediante un incremento de pixeles a su alrededor y además permite rellenar hoyos dentro del objeto. El número de pixeles añadidos depende del tamaño elemento estructurante.

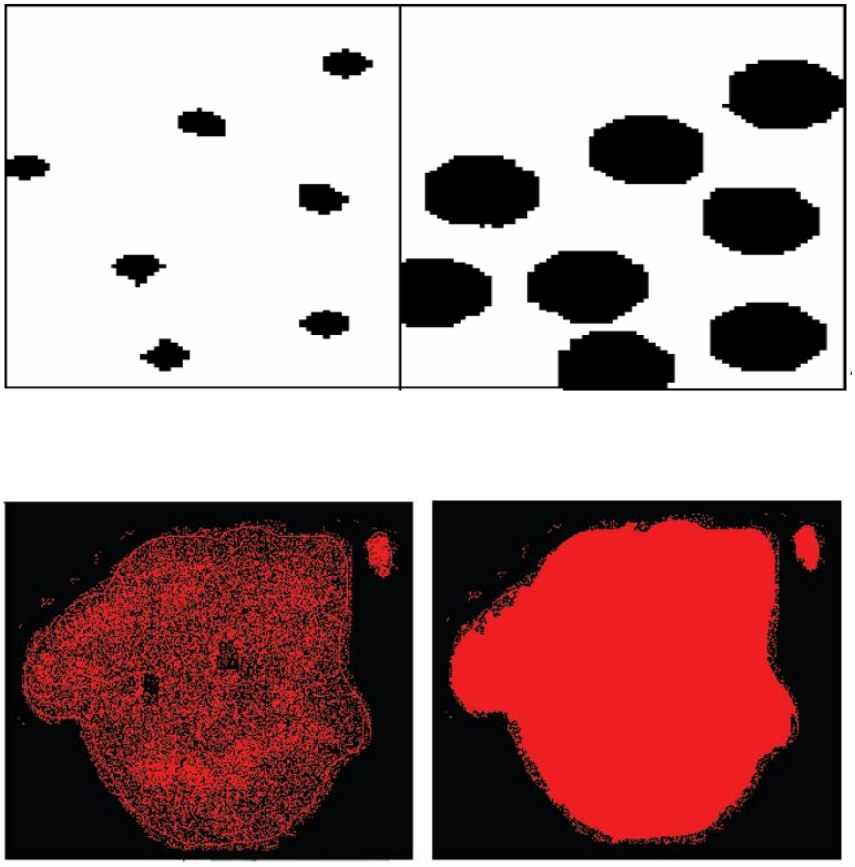


Figura 21. Ejemplos de imágenes dilatadas

- **Erosión o Cerrado**

Una operación de cerrado disminuye el tamaño de los objetos para esto remueve pixeles a su alrededor. Es el proceso inverso de la dilatación con el

cual suaviza secciones del contorno, pero generalmente une separaciones estrechas y rellena aberturas. Mediante la erosión también se elimina pixeles aislados, es decir, la operación erosión elimina ruido de la imagen.

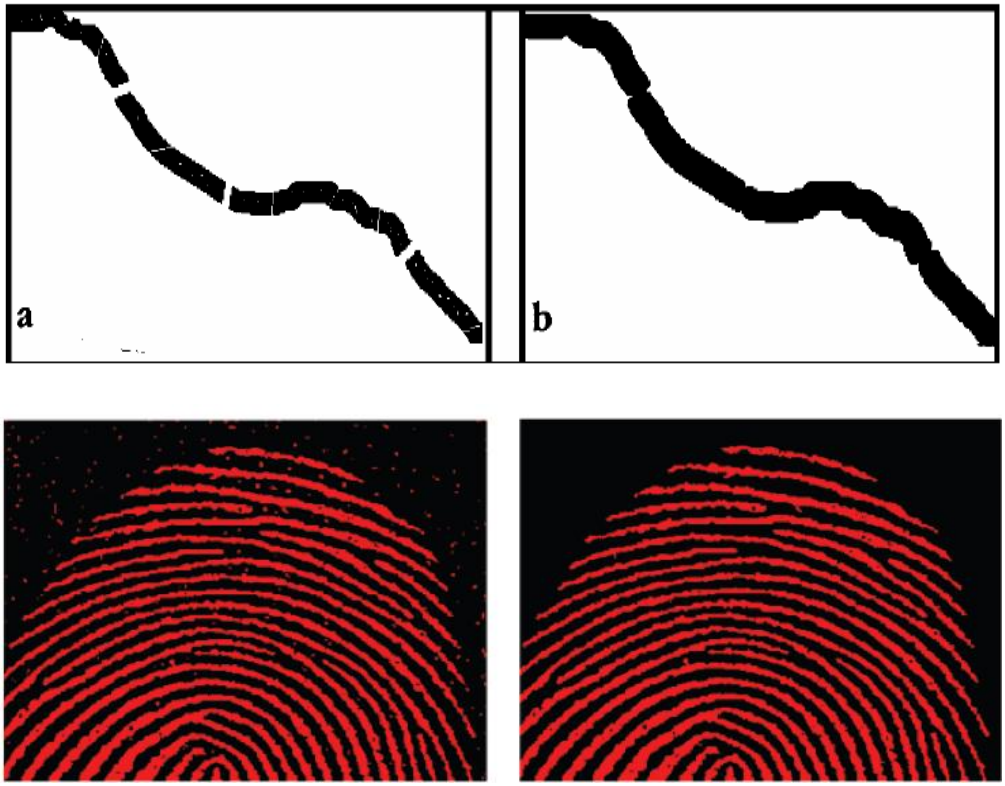


Figura 22. Ejemplos de imágenes erosionadas.

Segmentación basada en modelos (transformada de Hough)

Este algoritmo trata de detectar formas sencillas dentro de una imagen como líneas, circunferencias o cualquier tipo de curva usando ecuaciones analíticas de bordes conocidos. Para su ejecución requiere de una imagen binarizada en la que haya detectado lo bordes. El mayor inconveniente es su alto coste computacional.

Cuando no hay expresión analítica en las curvas a detectar, se emplean las transformadas de Hough generalizadas. El método es esencialmente un detector de formas o curvas geométricas.

Una de las características esenciales de la transformada de Hugh es la inmunidad frente al ruido.

A continuación se muestran ejemplos de la transformada de Hough.

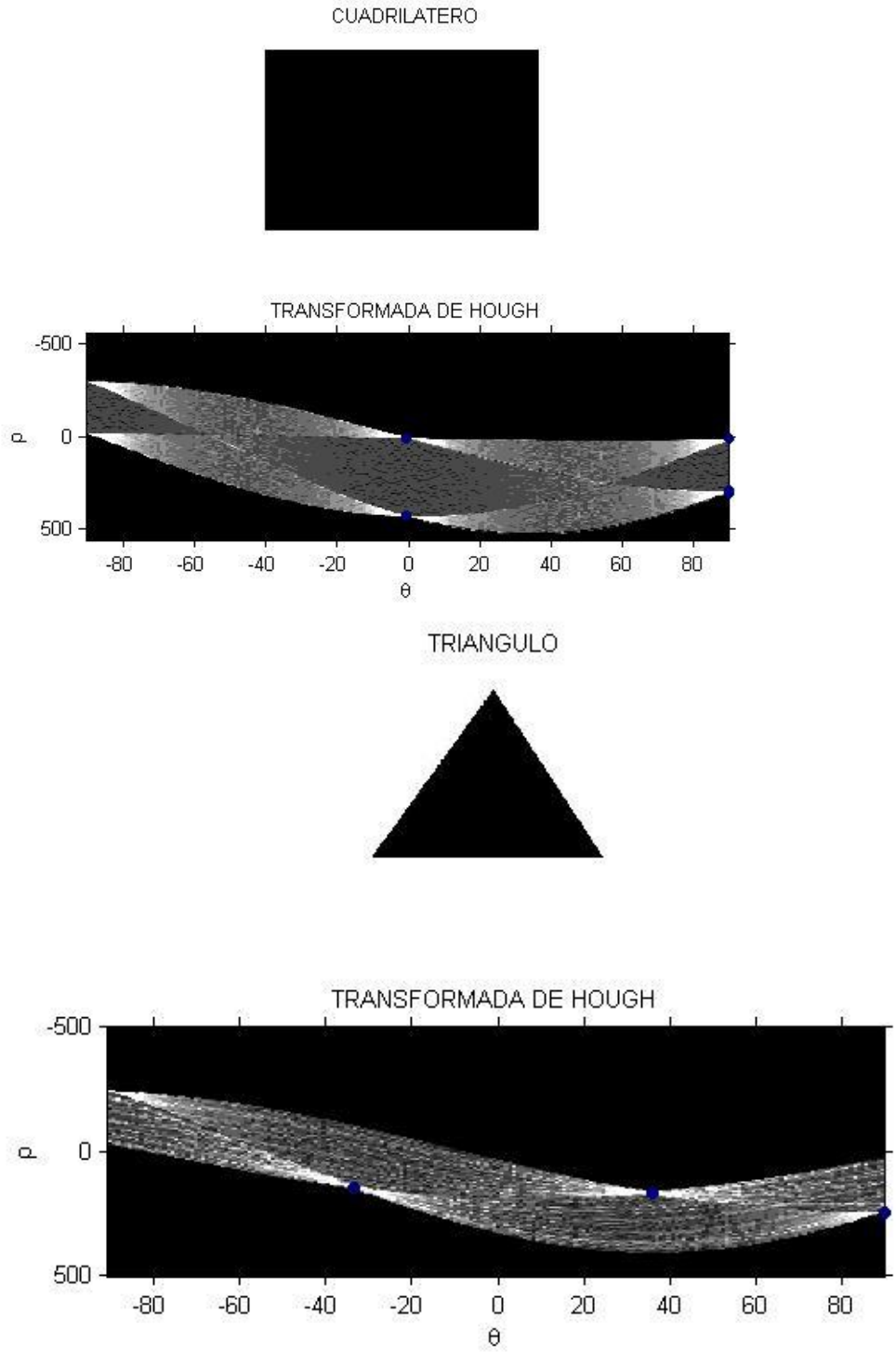


Figura 23. Ejemplos de la Transformada de Hugh.

1.4.3.4 Representación y Descripción

En esta etapa toda la información obtenida en la segmentación es analizada con el fin de extraer los rasgos que representan alguna información cuantitativa de interés o características básicas que nos permitan identificar un objeto de otro. Estos rasgos permiten describir una propiedad geométrica o estructural como por ejemplo: radio, área, textura, color, perímetro, distancias, etc.

Los algoritmos de análisis que se usan para aquí dependen de la específicamente aplicación a desarrollar, por lo cual no existe una manera de cómo generalizar esta etapa, una de las operaciones que comúnmente se realiza es la medición de objetos.

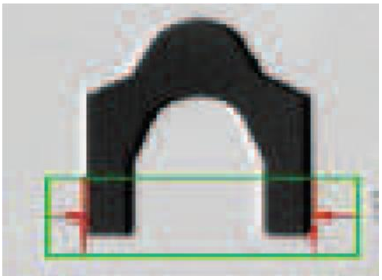


Figura 24. Medición de objetos.

1.4.3.5 Reconocimiento e Interpretación

Toda la información obtenida en la etapa anterior es utilizada aquí para de esta manera asignar etiquetas o nombres a un objeto en base a patrones o descriptores. La interpretación involucra la asignación de significado a un conjunto de objetos reconocidos.

En esta etapa del procesamiento de imágenes, se desarrollan algoritmos computacionales que simulen la visión humana para que permitan reconocer objetos, entender escenas y tomar decisiones o efectuar tareas. En la siguiente figura se puede observar la técnica de reconocimiento de objetos mediante patrones o coincidencias.

Figura 25. Búsqueda de objeto.

1.5 RECONOCIMIENTO DE VOZ

1.5.1 Definición de voz

La voz es un sonido emitido por un ser humano que se produce en el aparato fonador, en la laringe, cuando el aire procedente de los pulmones es forzado a través de la glotis, haciendo vibrar las cuerdas vocales, las cuales son más largas y gruesas en el hombre que en la mujer y el niño.

1.5.2 Cualidades de la voz

La voz posee tres cualidades importantes: tono o altura, intensidad y timbre, que van a ser descritas a continuación.

A. Tono o altura

Se refiere al rango tonal o de frecuencia que le corresponde al individuo, según sexo y edad. La misma frecuencia de vibración da siempre un mismo tono, independientemente de las otras cualidades del cuerpo vibrante. Cuanto mayor es la frecuencia, más agudo es el tono y al revés. Dentro de este concepto, aparece el de Frecuencia Fundamental (Fo), que es el resultado de la vibración de los pliegues vocales. Cada individuo presenta una frecuencia fundamental propia que va descendiendo con la edad en el caso de las mujeres y en el caso de los hombres va ascendiendo.

B. Intensidad

La intensidad o volumen de la voz, es la acción espiratoria de la respiración, es decir, el aire que sale desde los pulmones. La amplitud de vibración es la

que da la sensación de intensidad, viene dada por la presión aérea espiratoria y puede ser también disminuida o aumentada. La intensidad es medida en decibeles (dB). A continuación se presentan los niveles de intensidad de la voz humana.

C. Timbre

El timbre es la cantidad de armónicos que se forman al son de las frecuencias de los sonidos que se van emitiendo. Puede tener características diversas desde vivaz, estridente, monótono, pobre en armónicos, etc., esto estaría de alguna forma relacionado con el aspecto temperamental de cada persona.

Para que la frecuencia de la voz sea comprensible se deberá estar entre 500 y 3500 Hz y se requiere la presencia de armónicos, de hecho se han encontrado espectros conteniendo hasta 35; por otro lado, la energía de voz está contenida en su mayoría en las bajas frecuencias.

D. Formantes

El tracto vocal tiene cuatro o cinco resonadores llamados formantes. La frecuencia del formante es determinada por la forma del tracto vocal. Si el tracto vocal es un perfecto cilindro cerrado a nivel de la glotis y abierto al nivel de los labios y tiene una longitud de 17,5 cm, media aproximada de una laringe de hombre adulto, los primeros cuatro formantes estarán cerca de los 500, 1.500, 2.500 y 3.500 Hz. Agregando o acortando el tracto vocal, estas frecuencias básicas serán más graves o agudas; sin embargo, hay tres instrumentos para cambiar la forma del tracto vocal. La frecuencia de un formante en particular se puede cambiar de una dirección a otra. De acuerdo con Sundberg (1977), estos instrumentos son la mandíbula, el cuerpo de la lengua y la punta de la lengua

1.5.3 Principales aplicaciones del reconocimiento de la voz

El procesamiento digital de señales de voz tiene una gran variedad de aplicaciones, existe una base para el tratamiento digital de señales, que puede ser implementada para lograr obtener lo que nos interese según la aplicación.

En este caso, y como se comentó con anterioridad será utilizado para identificar ordenes que comandarán acciones aplicadas al campo de la domótica como el encendido de las luces de la casa, o artefactos eléctricos o incluso activación de alarmas.

- Aplicaciones locales
- Respuesta vocal interactiva
- Automatización de sistemas de encendido y regulación de componentes electrónicos.
- Automatización de sistemas telefónicos

1.5.4 Caracterización de los sistemas de reconocimiento de voz

Característica esencial a definir en el proceso de captura de la señal de voz es la frecuencia de muestreo, esto es causante de la diferencia entre una buena calidad de señal.

Factores a analizar según (Kirschning 1998)

- Tamaño del vocabulario y confusión
- -Sistemas dependientes e independientes del locutor
- Voz aislada, discontinua y continua
- Voz aplicada a tareas o en general
- Voz leída o espontanea
- Condiciones adversas

1.5.5 Variabilidad de las señales de Voz

Algunas fuentes de variabilidad incluyen los siguientes aspectos:

- Variabilidad en un locutor en mantener una pronunciación consistente y en el uso de palabras y frases.
- Variabilidad entre locutores debidos a diferencias fisiológicos
- Variabilidad entre transductores cuando se habla frente a diferentes micrófonos o aparatos telefónicos.
- Variabilidad introducida por el sistema de transmisión (redes de comunicación teléfonos celulares, etc.).
- Variabilidad en el ambiente, que incluyen conversaciones extrañas y eventos acústicos de fondo, como ruidos, etc.

1.5.6 Sistemas de reconocimiento de voz y técnicas empleadas

Se conoce como reconocimiento al sistema computacional capaz de procesar la señal emitida por un ser humano y convirtiéndola en órdenes, imágenes o texto, en otras palabras permite la comunicación entre seres humanos y computadoras.

Técnicas

A. Comparación de plantillas o patrones.- El método de Plantilla o Patrones utiliza técnicas de Programación Dinámica (DTW) y básicamente consiste en comparar el patrón a reconocer (de entrada) con una serie de plantillas o patrones que representan a las unidades a reconocer.

B. Modelos ocultos de Markov (HMM)

El modelado estocástico de la señal de habla soluciona el problema que presentaba la técnica de alineamiento de plantillas, proporcionando los mejores resultados para el reconocimiento de habla aislada como continua y para independencia del locutor.

C. Redes neurales o neuronales (NN)

Las redes neuronales son estructuras de procesamiento paralelo de información, formadas por numerosos nodos simples conectados entre sí mediante pesos y agrupados en diferentes capas, entre las que se deben distinguir la capa de entrada y la capa de salida. Debido a su naturaleza intrínsecamente no lineal, a su capacidad de clasificación, y sobre todo a la capacidad que tienen para aprender una determinada tarea a partir de pares observación-objetivo sin hacer suposición alguna sobre el modelo subyacente, se han convertido en una de las herramientas más atractivas para la solución del problema del reconocimiento de habla.

1.5.7 Reconocimiento automático de voz

El reconocimiento consiste básicamente de dos pasos:

Primer paso: segmentación y rotulado. La señal es dividida en regiones acústicas a las que son asignados uno o más fonemas, resultando en una caracterización de la señal de voz mediante un reticulado de fonemas.

Segundo paso: se trata de determinar una palabra (o conjunto de palabras) válida a partir de la secuencia de fonemas rotulados en el primer paso. Se introducen en esta etapa restricciones lingüísticas (vocabulario, sintaxis, y reglas semánticas)

La primera etapa en el procesamiento (que es común a todos los enfoques) es la etapa de análisis de voz, que provee una representación (espectral) de las características no estacionarias de la señal de voz.

En la siguiente etapa es la extracción de característica en donde se convierten las medidas espectrales en un conjunto de parámetros que describen las propiedades acústicas de las unidades fonéticas.

La tercer etapa del procesamiento es la etapa de segmentación y rotulado en donde el sistema trata de encontrar regiones estables donde las

características cambian poco, que son rotuladas teniendo en cuenta cuán bien la característica en la región se ajusta a unidades fonéticas individuales.

1.5.8 Reconocimiento de patrones

Consiste básicamente en dos pasos:

Primer Paso: entrenamiento de patrones

Segundo Paso: comparación de patrones

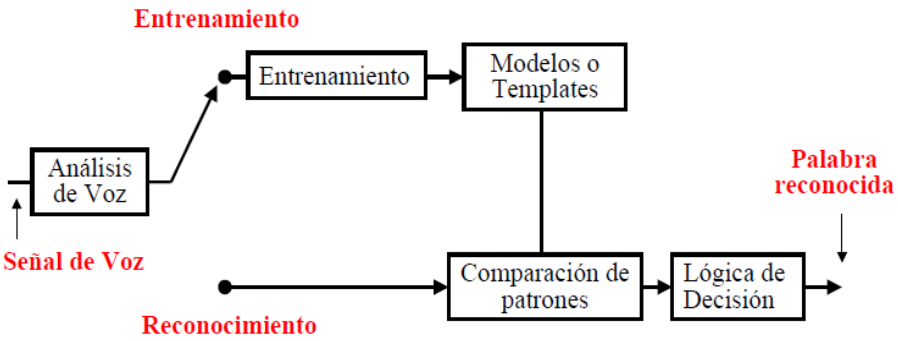


Figura 26. Reconocimiento de Voz basado en reconocimiento de patrones.

1.5.9 Comandos de voz

El objetivo es la implementación de un sistema que con el reconocimiento del habla permita, controlar dispositivos eléctricos, destinados a discapacitado los cuales requieren una ayuda extra para realizar tareas cotidianas dentro de la vivienda.

Para ello se debe hacer un estudio de la voz, sus características y técnicas de parametrización, además se implementara el hardware que será el encargado de enviar la señal a los elementos actuadores.

El sistema debe reconocer la palabra clave que constituye el comando de voz y no quien la dice, para la interfaz con la pc se puede utilizar el puerto serial EIA RS232.

1.5.10 Etapas para el reconocimiento de la voz

Es necesario conocer las características y el tratamiento que se le debe dar, en el caso de reconocimiento del habla se debe seguir el siguiente esquema

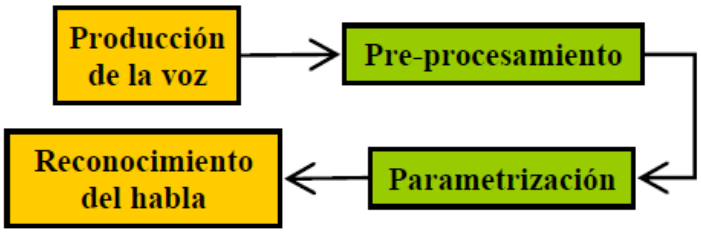


Figura 27. Etapas de procesamiento de la voz

1.5.10.1 Pre- procesamiento de la señal de voz.

Esta etapa aplicado a la señal de voz corresponde a los pasos previos a la parametrización, con lo cual se puede resaltar sus características más importantes y después poder analizar con alguna técnica de parametrización.

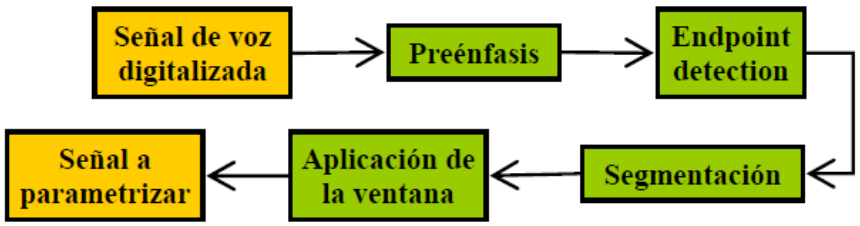


Figura 28. Parametrización

1.5.10.2 Preenfasis.- Se realiza para hacer el procesamiento de la señal menos susceptible a truncamientos, aplanarla espectralmente y para compensar la caída de 6 dB que experimenta la seña al pasar a través del tracto vocal[9].

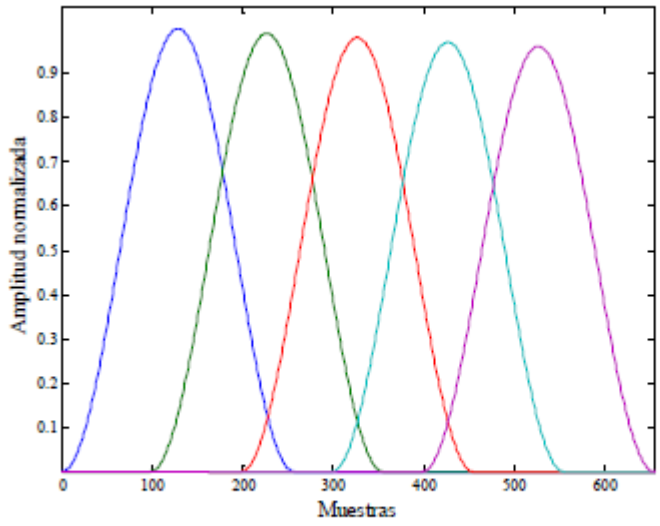
[9] Kornhauser, D. (4 de Marzo de 1999). *Diseno de un reconocedor de voz de palabras aisladas publico.* Recuperado el 15 de Febrero de 2013, de http:bq.unam.mx/-daniel/gvoice/tesis/node22.html

1.5.10.3 Endpoit Detection.- Luego de tener la señal grabada en el computador es necesario la determinación del comienzo y final de la parte útil de la señal, que mediante un algoritmo busca los puntos de la señal donde se concentra la energía y extrae solo ese fragmento[10].

1.5.10.4 Segmentación y aplicación de la ventana.

Una vez obtenida la información útil de la señal de voz, se hace necesario dividir la señal en tramas de N muestras, donde N es un valor que se escoge tomando en cuenta que la señal de voz es estacionaria a "trozos", condición necesaria para poder realizar el análisis de Fourier en tiempo corto.

Por lo general se utilizan tramas de 256 muestras por ser un valor que establece un equilibrio entre la resolución en tiempo y en frecuencia para una señal de voz.



Segmentación de la señal de voz en tramas solapadas entre ellas

Figura 29. Segmentación

Luego se debe escoger el tipo de ventana que se le va aplicar a cada trama, su elección es importante para analizar el efecto de la misma sobre la

[10] Slavinshy, J.(1999). Speaker Verification. Extraido el 20 de marzo de 2013 desde http://www.owlnet.rice.edu/~elec301/Projects99/wrcocee/endpt.htm

resolución espectral de la señal, la cual depende del ancho del lóbulo principal de la venta y la atenuación de los lóbulos secundarios respecto al principal, lo ideal es que el lóbulo principal sea muy estrecho y los lóbulos secundarios sean lo más pequeño posible.

En el caso de la ventana de Hamming se cumple con el requerimiento de que la atenuación de los lóbulos secundarios sea brusca, a pesar de que su lóbulo principal es más ancho que para la ventana rectangular, lo que permite tener mejor resolución espectral.

L a ventana de hamming está definida por la siguiente ecuación.

$$W\ nT\ = 0{,}54 - 0{,}46\cos\ 2.\frac{n}{N} \quad Donde\ 0 < n < N$$

1.5.10.5 Técnicas de parametrización de voz

El objetivo principal de esta representación es la de comprimir los datos correspondientes a la señal de voz, eliminando información ajena al análisis fonético de la información y extraer características con contribuyan con la detección de las diferencias fonéticas y que no son apreciables mediante un análisis en tiempo o en frecuencia.

Entre ella tenemos

- Cepstrum
- Predicción lineal de espectro
- Técnicas basadas en el espectro de Fourier

1.5.10.6 Reconocimiento del habla

El proceso de parametrización genera coeficientes que representan características de la señal de voz, que pueden ser usadas en la etapa de reconocimiento del habla, el tamaño de la matriz obtenida del proceso de parametrización depende directamente de la longitud de la señal de voz, la cual tiene relación con la palabra y el hablante. Por lo cual se hace

necesaria la estandarización de la matriz que contiene los coeficientes cepstrales calculados, para que el tamaño de las matrices usadas para el reconocimiento del habla sea el mismo.

El proceso de estandarización de la matriz de coeficientes cepstrales es el primer paso para el reconocimiento del habla y se denomina Cuantización Vectorial. Y el siguiente paso es el cálculo de la diferencia entre la señal de voz hablante y las señales que se encuentran en la base de datos de entrenamiento del sistema, dicha diferencia se obtiene mediante el cálculo de la Distancia Euclidiana en varias dimensiones.

1.5.10.7 Cuantificación Vectorial .- La finalidad de la cuantización Vectorial es la de obtener una matriz que represente de la manera más exacta a la matriz original y que además permita hacer una comparación razonable entre la señal parametrizada del hablante y las señales contenidas en la base de datos del sistema de reconocimiento de habla, ya que todas las matrices tienen el mismo tamaño y es posible emplear una técnica como la distancia euclidiana para determinar la diferencia entre las mismas y realizar el proceso de identificación de la palabra.

La forma de medir la fidelidad de un cuantificador es determinar el error que este produce al reemplazar los datos de entrada que recibe por los vectores representantes o codewords, dicho parámetro es llamado error por distorsión.

La finalidad de un cuantificador es obtener un conjunto de vectores representativos llamado “codebook”, que presente el menor error por distorsión, por ejemplo para cuantificar los vectores de observación.

Componentes de un cuantificador vectorial

- Vectores de observación

- Clasificación de Vectores

Algoritmos de Clasificación:

Los principales algoritmos de clasificación de vectores son descritos a continuación:

- Algoritmo K-Means
- Algoritmo LBG

1.5.10.8 Distancia Euclidiana.- La distancia Euclidiana se emplea, para el reconocimiento del habla, como método para calcular la diferencia existente entre el "codebook" obtenido de la palabra dicha por el hablante y el resto de "codebooks" almacenados en la base de datos de entrenamiento del sistema. El resultado final de dicha comparación es un valor numérico que representa la distancia entre dos matrices de iguales dimensiones. El "codebook" perteneciente a la base de datos de entrenamiento del sistema cuya distancia al "codebook" generado para representar la palabra dicha por el hablante sea menor que el nivel de comparación establecido por el programador de la aplicación, identifica la palabra con la que ya existe mayor semejanza. La fórmula matemática empleada para el cálculo de la distancia euclidiana en dos o más dimensiones se muestra en la siguiente ecuación:

$$d\ v,w\ = \sqrt{v_1 - w_1\ ^2 + \ v_2 - w_2\ ^2 + \ v_3 - w_3\ ^2 + \cdots + \ v_n - w_n\ ^2}$$

$$Donde\ v = \ v_1\ v_2\ v_3\ \ldots v_n\ \ y\ w = \ w_1\ w_2\ w_3\ \ldots w_n\ \ son\ matrices\ de\ logitud\ n$$

Se acostumbra usar dicha métrica para el cálculo de la distancia entre dos puntos, pero su aplicación va más allá, permitiendo calcular la distancia entre dos matrices.

1.5.11 Componentes y proceso utilizado para el caso práctico de reconocimiento de voz

Basados en la teoría antes estudiada, podemos resumir que los componentes y el proceso a seguir para la obtención de la señal y su debido tratamiento para con ello lograr manipular el accionar de aparatos electrónicos mediante un comando de voz, que para ello no se necesita

identificar quien hace la orden, solamente se debe reconocer la acción que se quiere ejecutar.

El proceso a seguir es:

- Toma de la señal a través de un micrófono
- Adquisición de la señal en un software que permita su tratamiento. (Matlab u otros)
- Procesamiento de la señal
- Detección y eliminación de los periodos de silencio
- Parametrización
- Reconocimiento de voz
- Interfaz grafica

1.6 ANDROID

1.6.1 Introducción

Desde un tiempo atrás los usuarios de móviles han ido requiriendo de más funciones que permitan satisfacer sus necesidades. Por esta razón las compañías fabricantes de teléfonos ofrecen día a día mejoras en sus dispositivos llegando al punto de que estos son capaces de realizar muchas funciones destinadas, antiguamente, exclusivamente para las computadoras. Google no ha desaprovechado la oportunidad de realizar junto con su gran equipo de desarrolladores un sistema operativo moderno y a la altura de las circunstancias de nominado Android SO para teléfonos y tabletas.

Android es un sistema operativo además de una plataforma de software basa en el núcleo de Linux. Diseñada en un principio para dispositivos móviles. Android permite controlar dispositivos por medio de bibliotecas desarrolladas o adaptadas por Google mediante el lenguaje de programación Java.

1.6.2 Historia

Fue desarrollado por Android inc, empresa comprada por Google en 2005. Inicialmente el sistema fue desarrollado exclusivamente por Google pero

más tarde pasó a formar parte de la Open Handset Alliance, un consorcio de 78 compañías de Hardware, Software y telecomunicaciones, las cuales llegaron a un acuerdo para promocionar los estándares de códigos abiertos para dispositivos móviles. Una de las ventajas que contiene el Android es que permite que personas ajenas a Google desarrollen aplicaciones, así consiguen que las aplicaciones basadas en el sistema sean mucho mayores que las de los sistemas cerrados, y esto repercute en la venta de terminales con Android SO.[11]

El primer teléfono que salió al mercado con este dispositivo fue el HTC G1 o Dream lanzado originalmente en EEUU a finales de 2008, venía con la versión Android 1.0 instalada y rápidamente se convirtió en un éxito de ventas, actualmente las últimas versiones poco tienen que ver con el anticuado sistema 1.0.[12]

1.6.3 Características[13]

Las características más relevantes son:

- Diseño de dispositivo, Android permite la utilización de casi cualquier resolución móvil disponible hasta día de hoy lo que hace el sistema adaptable desde pantallas de 2,5" hasta tabletas de 10" .Además permite el uso de librerías de gráficos en 2D y 3D.
- Almacenamiento, se realiza mediante SQLite, una base de datos liviana, que es usada para propósitos de almacenamiento de datos.
- Conectividad, Android soporta las siguientes tecnologías de conectividad: GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE, HSDPA, HSPA+ y WiMAX.
- Mensajería, SMS y MMS son formas de mensajería, incluyendo mensajería de texto y ahora la "Android Cloud to Device Messaging

[11] http://es.wikipedia.org
[12] Enrique Ramírez Hernández. **"***Desarrollo de Aplicaciones para Dispositivos con Sistemas Operativo Android***"**. Valencia, 2011.
[13] http://es.wikipedia.org

Framework (C2DM)" es parte del servicio de PushMessaging de Android.

- Navegador Web, basado en el motor de renderizado de código abierto WebKit, emparejado con el motor JavaScript V8 de Google Chrome. El navegador por defecto de Ice CreamSandwich.
- Soporte Java, realmente Android no dispone de una maquina virtual de java como tal, debido a que los dispositivos móviles son de escasos recursos así que se apoya en la máquina virtual de Dalvik para ejecutar el código java.
- Soporte multimedia, Android soporta los siguientes formatos multimedia: WebM, H.263, H.264 (en 3GP o MP4), MPEG-4 SP, AMR, AMR-WB (en un contenedor 3GP), AAC, HE-AAC (en contenedores MP4 o 3GP), MP3, MIDI, OggVorbis, WAV, JPEG, PNG, GIF y BMP.
- Soporte para streaming, el soporte para streaming de descarga en HTML se hace de forma nativa, pero el soporte para adobe flash se hace mediante una aplicación que proporciona Adobe, aunque en dispositivos actuales se está empezando a ejecutar aplicaciones en Adobe Flash.
- Soporte para hardware adicional, el sistema soporta una infinidad de software externo, entre estos encontramos: cámaras de fotos, de vídeo, pantallas táctiles, GPS, acelerómetros, magnetómetros, giroscopios, aceleración 2d y 3d, sensores de proximidad, presión luz, gamepad, termómetro.
- Entorno de desarrollo, el entorno de desarrollo integrado es Eclipse usando el plugin de Herramientas de Desarrollo de Android y un emulador suministrado por Google.
- Google Play, es un catálogo de aplicaciones gratuitas o de pago en el que pueden ser descargadas e instaladas en dispositivos Android sin la necesidad de un PC.

- Multi-táctil, se incluyen estas funciones de forma nativa para todos los dispositivos Android que dispongan de una pantalla con esta capacidad.
- Bluetooth, la versión 2.2 ya da soporte para A2DF y AVRCP, envío de archivos (OPP), exploración del directorio telefónico, marcado por voz y envío de contactos entre teléfonos.
- Video llamada, Android soporta video llamada a través de Google Talk desde su versión HoneyComb.
- Multitarea, para todas las aplicaciones y desde su primera versión.
- Características basadas en voz, la búsqueda en Google a través de voz está disponible como "Entrada de Búsqueda" desde la versión inicial del sistema.
- Tethering, permite al teléfono ser usado como punto de acceso WIFI alámbrico o inalámbrico, está disponible a partir de la versión 2.2.

1.6.4 Arquitectura[14]

Los componentes principales de un sistema Android son los siguientes:

- **Aplicaciones**: las aplicaciones básicas que se incluyen en el sistema son: correo electrónico, programa de SMS, calendario, mapas, navegador, contactos, etc. Todas las aplicaciones están escritas en lenguaje de programación Java

- **Framework de aplicaciones:** los desarrolladores tienen acceso completo a los mismos APIs del framework usados por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes.

- **Bibliotecas:** Android incluye un conjunto de bibliotecas de C/C++ usadas por varios componentes del sistema, las mismas que se exponen en el framework de aplicaciones para que los

[14] Raúl Cano Calabuig. **"***Diseño e implementación de una aplicación domótica en Android reconfigurable a partir de un XML***".** Valencia, 2010

desarrolladores hagan uso de ellas. Algunas son: System C library (implementación biblioteca C estándar), bibliotecas de medios, bibliotecas de gráficos, 3D y SQLite, entre otras.

- **Runtime de Android**: Android incluye un set de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las bibliotecas del lenguaje Java. Cada aplicación Android corre su propio proceso, con su propia instancia de la máquina virtual Dalvik. Dalvik ha sido escrito de forma que un dispositivo puede correr múltiples máquinas virtuales de forma eficiente.

- **Núcleo Linux**: Android depende de Linux para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red y modelo de controladores. El núcleo también actúa como una capa de abstracción entre el hardware y el resto de la pila de software.

CAPITULO 2

SISTEMA DE CONTROL DOMOTICO MEDIANTE VISIÓN ARTIFICIAL

2.1 SISTEMA DE VISIÓN ARTIFICIAL

Con la finalidad de ayudar a la personas con discapacidad a tener un mayor grado de autonomía y confort dentro de su hogar se seleccionó diseñar un sistema de control mediante visión artificial, el cual permita gestionar cargas básicas como son: iluminación, puertas, ventanas, ventiladores. Además se hará una aplicación que permita dar información de obstáculos ubicados al frente del usuario, permitiendo de alguna manera ayudar al discapacitado a tomar decisiones para su correcto desplazamiento en el interior de su vivienda.

La visión artificial tiene un gran campo de aplicación y es utilizada actualmente en muchos procesos ya sea industriales, científicos e incluso en la medicina. Por lo que la idea de hacer sistema de control domótico basado en este campo es factible.

Este sistema de control consiste básicamente en reconocer señales realizadas con las manos y cada señal corresponde a una acción sobre las cargas a controlar. En este proyecto se utilizó el software de “Labview” y su “ToolKi” de visión artificial “IMAQ Vision” de "National Intruments”.

2.2 DESCRIPCIÓN DEL SOFTWARE

2.2.1 NI Labview

Es un software muy versátil cuya principal ventaja, es que puede ser usada tanto por programadores expertos, como personas con pocos conocimientos

en el tema. Puesto que Labiew usa el lenguaje G (Lenguaje gráfico de programación) que es relativamente fácil de comprender.

Aunque es necesario aclarar que para desarrollar programas que se apliquen a la automatización, control, adquisición y manejo de datos sí es necesario tener conocimientos más avanzados no solo de programación, sino de otras áreas específicas para cada aplicación del programa que se tenga planeado.

LabVIEW es principalmente utilizado por los ingenieros para el manejo de datos, la comunicación entre una computadora y un aparato o circuito externo es imprescindible para las aplicaciones que se le pueden dar al software, por lo que Labview puede comunicarse con interfaces como:

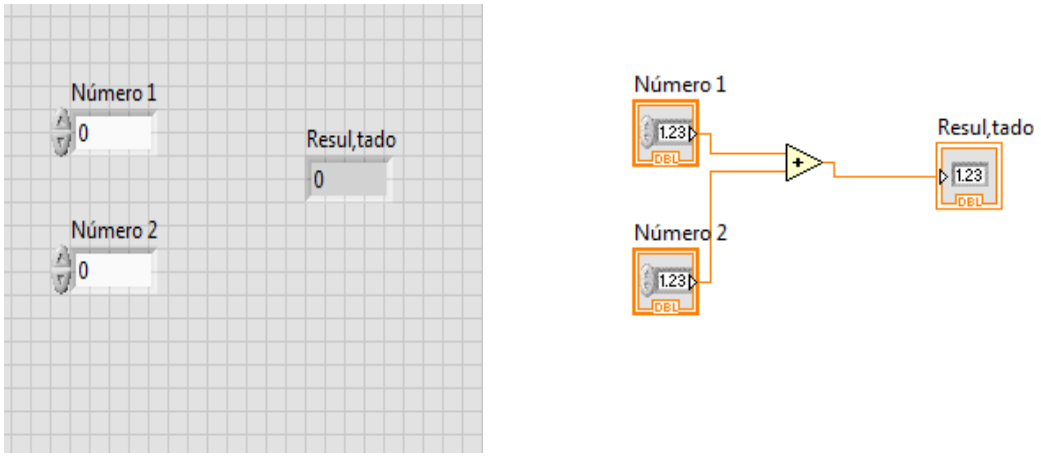
- Puerto serial
- Puerto paralelo
- GPIB
- PXI
- VXI
- TCP/IP
- Irda
- Bluetooth
- USB
- OPC

Entre otros.[15]

Los programas realizados en Labview se denominan instrumentos virtuales "Vis", dado que tienen la apariencia de un instrumento real. Estos "Vis" están conformados por dos partes: el panel frontal y el diagrama de bloques. El panel frontal es la interfaz usuario, ya que se pueden ingresar datos y

[15] *Manual Básico de programación en Labiew* de MarterHacks. 83p.

visualizar resultados. En el diagrama de bloque se encuentra todo el código fuente de VI.[16]



(a) (b)

Figura 30. Panel Frontal, b) diagrama de Bloques.

2.2.2 NI IMAQ

NI IMAQ es un paquete de funciones que controla el hardware de adquisición de imágenes y además contiene métodos de para ejecutar tareas, que van desde la inicialización del hardware, adquisiciones simples y secuenciales hasta adquisiciones avanzadas a alta velocidad. [17]

2.2.3 IMAQ Vision

"IMAQ Vision "es una librería para LabView que permite implementar aplicaciones inteligentes de visión artificial, presenta herramientas complejas de procesamiento digital de imágenes como los detectores de borde y reconocimiento de patrones complejos entre otros.[18]

En "IMAQ Vision", los VIs se encuentran organizados en tres partes:

- **Vision Utilities:** permite adquirir, manipular y mostrar características de las imágenes.

[16] HOLGÍN, German; PERÉZ, Sandra y OROZCO, Álvaro. *Curso básico de Labiew 6i*. Pereira: Universidad Tecnológica de Pereira, 2002, 265p.
[17] *National Instruments*. Ni-IMAQ Users Manual. Austin. Texas. 2003. 112p
[18] http://www.csun.edu/~rd436460/Labview/IMAQ-Manual.pdf

- **Image Processing:** permite analizar, filtrar y procesar las imágenes adquiridas.

- **Machine Vision:** permiten realizar tareas comunes de visión artificial como: conteo de objetos, detección de bordes, comparaciones de patrones, medición de ángulos, distancias, etc.

2.2.4 Vission Assistant

Este asistente permite experimentar con diferentes funciones de visión, y de esta manera determinar el mejor procedimiento que se adecue para la aplicación a desarrollar. Con esto se puede crear una aplicación eficiente y confiable. Una vez que se determina cómo resolver de mejor manera la aplicación el asistente genera automáticamente el código para ser ejecutado en Labview. Sin embargo luego de ejecutar el asistente se debe agregar al código algunos elementos adicionales para mejorar la aplicación.

El asistente tiene acceso a todas las librerías, de Labview, de NI IMAQ o de "NI IMAQ Vision" por lo que facilita enormemente la elaboración de la aplicación.

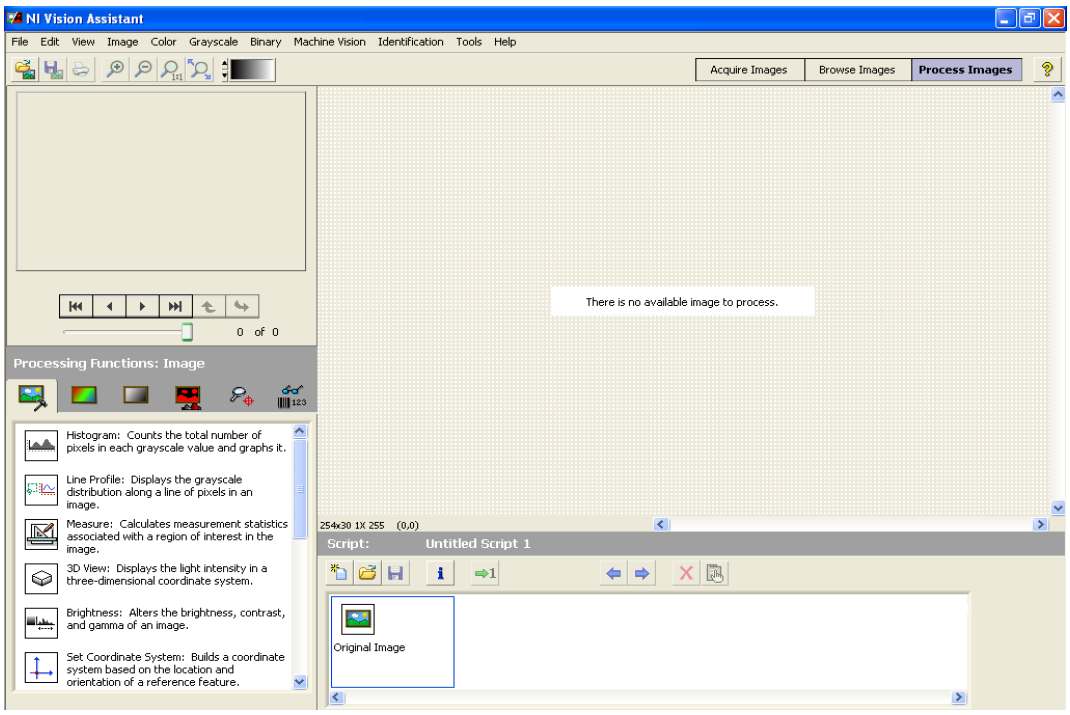


Figura 31. Ventana del asistente de visión.

2.2.5 Measurement & Automation Explorer (Max)

Es un explorador que permite realizar la configuración de todo software y hardware de "National Intruments" instalado en el equipo. Mediante MAX se pueden crear y editar nuevas interfaces, tareas y canales para cada dispositivo.

El MAX ofrece una buena herramienta para modificar los parámetros de operación del hardware con el fin de mejorar su desempeño en una aplicación específica. En la siguiente figura se indica la ventana del MAX.

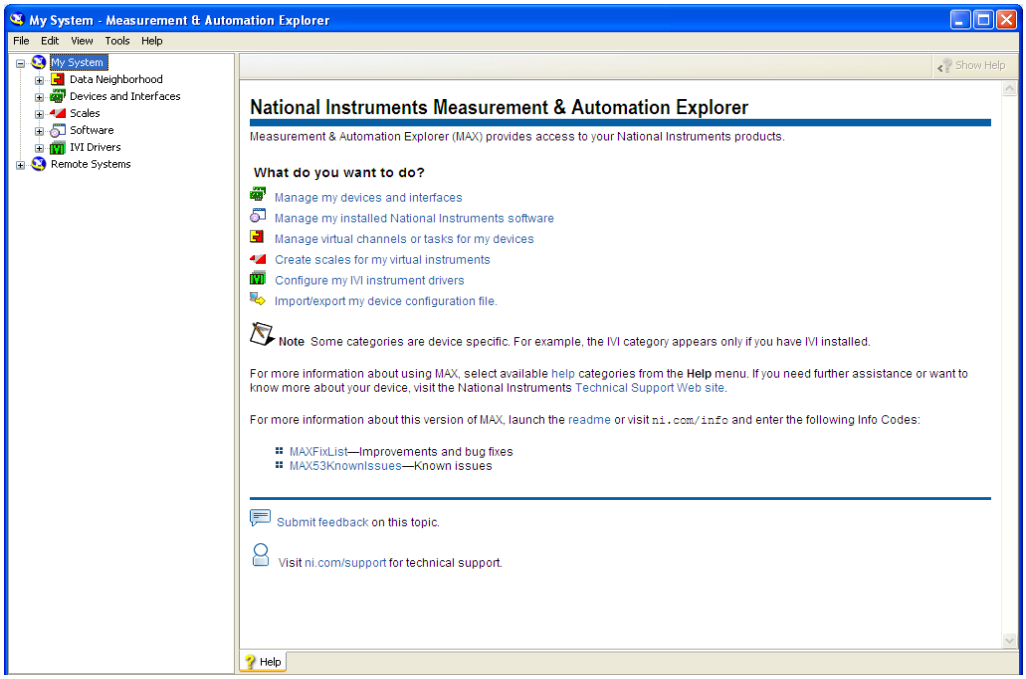


Figura 32. Ventana del explorador MAX.

2.2.6 Herramientas de Visión[19]

La visión artificial es comúnmente usada dentro de cinco aéreas de aplicación, las cuales incluyen:

Mejora de una imagen: utiliza herramientas de filtrado para afilar bordes, remover ruido, o extraer información.

[19] EDISON PAUL ORDOÑESZ ALQUINGA Y GLORIA ALEXANDRA QUILUPANGUI LUJE **"***Diseño e Implementacion de Clasificacion de Rosas Aplicando Vision Artificial con Labview***".** QUITO, 2011.

Revisión de presencia: para verificar la presencia de una parte o característica, se puede usar cualesquier herramienta de color, igualación de patrones o histograma.

Ubicación de características: la detección de bordes, igualación de patrones en escala de grises, igualación de forma, igualación geométrica e igualación de patrones de color, son herramientas que se pueden utilizar para localizar características.

Medición de características: la razón para usar un sistema de visión artificial es tomar una medida. Se puede usar herramientas de detección de bordes, análisis de partículas, y funciones geométricas para medir distancia, diámetro, ángulos y área.

Identificación de partes: la identificación de partes es importante para la conformidad, rastreo y verificación. Los métodos directos de identificación incluyen lectura de código de barras, y OCR.
En la siguiente figura se indica las herramientas y funciones que se usan para trabajar con sistemas de visión artificial.

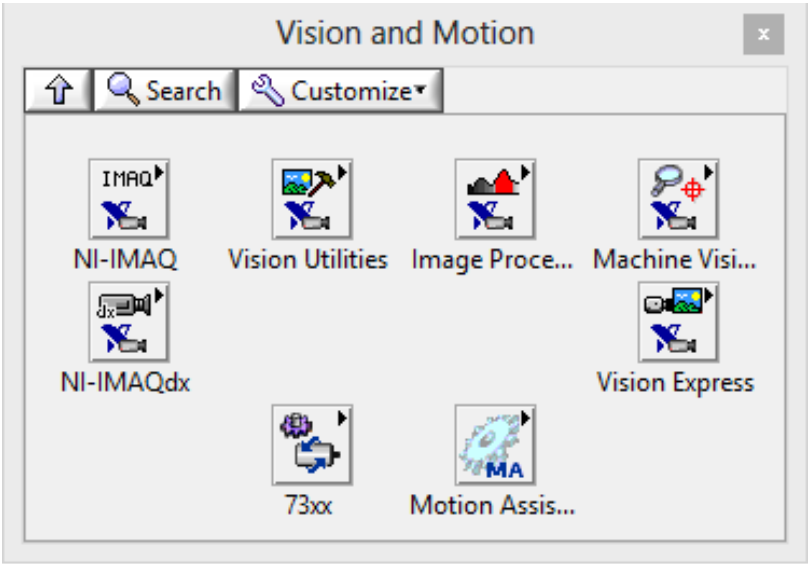


Figura 33. Herramientas para adquirir y procesar una imagen.

2.3 FUNCIONAMIENTO DEL SISTEMA

El sistema consiste en identificar señas producidas por las manos las cuales van a ser captadas por una cámara USB conectada a la pc. Y mediante un software realizado en Labview permite identificarlas. El software básicamente busca patrones previamente creados y al identificar alguno procede a ejecutar la acción correspondiente a dicha señal.

Este sistema puede controlar básicamente lo siguiente: encendido/apagado de la iluminación, ventiladores, apertura/cierre de puertas, ventanas.

Debido a que este sistema es parte de un prototipo de investigación y no un producto final. Para su funcionamiento le asignamos las mejores condiciones en cual es sistema trabaje sin mayores complicaciones.

2.4 ADQUISICION DE LA IMAGEN

Para la adquisición de la imagen se utilizó una cámara USB de la marca “Genius FaceCam 1000” la cual va colocada en la parte superior de una caja a la cual denominamos “caja de adquisición” cuya finalidad dar las mejores condiciones para obtener una imagen de calidad del objeto a analizar y de esta manera facilitar la etapa de procesamiento de la imagen.

Figura 34. Caja de adquisición de datos

2.5 DESARROLLO DEL SOTWARE EN LABVIEW

Como ya se mencionó anteriormente el software a desarrollar reconoce ciertas señas a las cuales se les atribuye una función de control. Y el primer paso en la realización del código es acceder a la cámara USB para adquirir las imágenes esto lo podemos hacer usando la paleta de herramientas de "NI-IMAQdx".

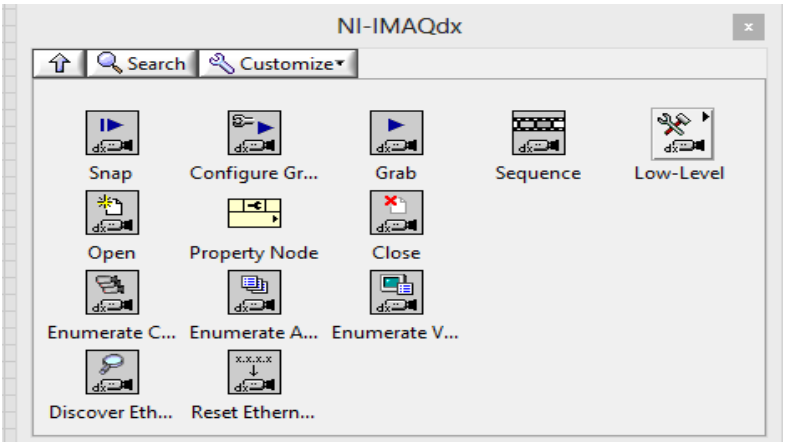


Figura 35. Herramientas de "NI-IMAQdx".

Y para obtener una adquisición continua de la imagen se debe realizar el siguiente código en Labview.

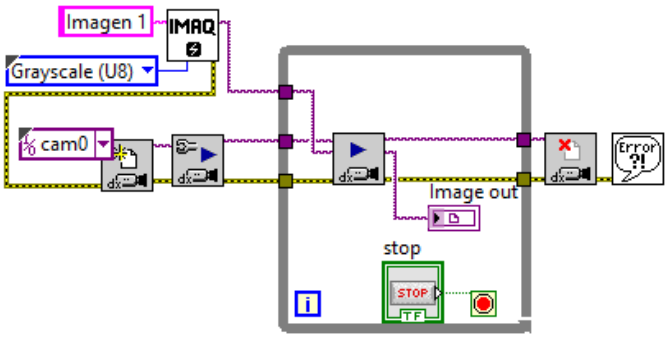


Figura 36. Código para adquisición continúo de imágenes.

Luego utilizamos el asistente de Visión denominado "Vission Assistant" que se encuentra dentro de la paleta de "Vision Express" para procesar las imágenes adquiridas.

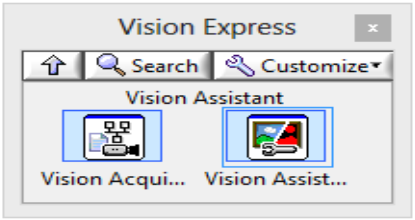


Figura 37. Paleta de "Vision Express".

La razón de su uso es la versatilidad que tiene para experimentar con las herramientas de visión artificial. Y por otra parte, en nuestro sistema únicamente se pretende identificar patrones para efectuar control y con el asistente de visión podemos probar cual es la mejor manera de realizarlo.

Una vez que damos click sobre el icono se abre el asistente en el cual podemos ir procesando la imagen.

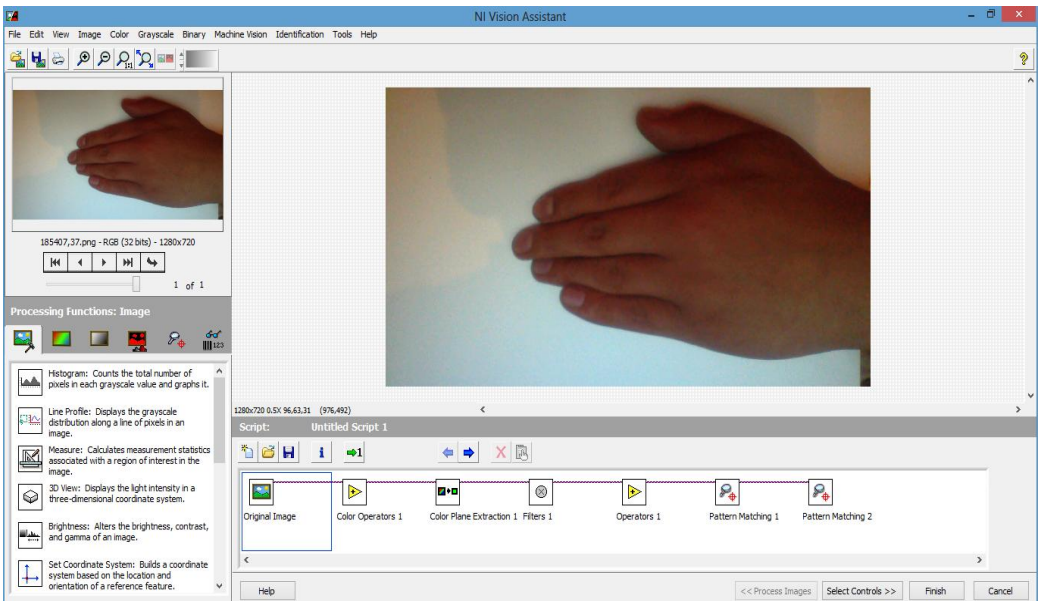


Figura 38. Asistente de visión Artificial.

En primera instancia tratamos de que el color blanco no resalte dentro de la imagen para esto realizamos una operación "not or" al color blanco, con esto el color blanco se atenúa resaltando el objeto. Como se puede mostrar a continuación.

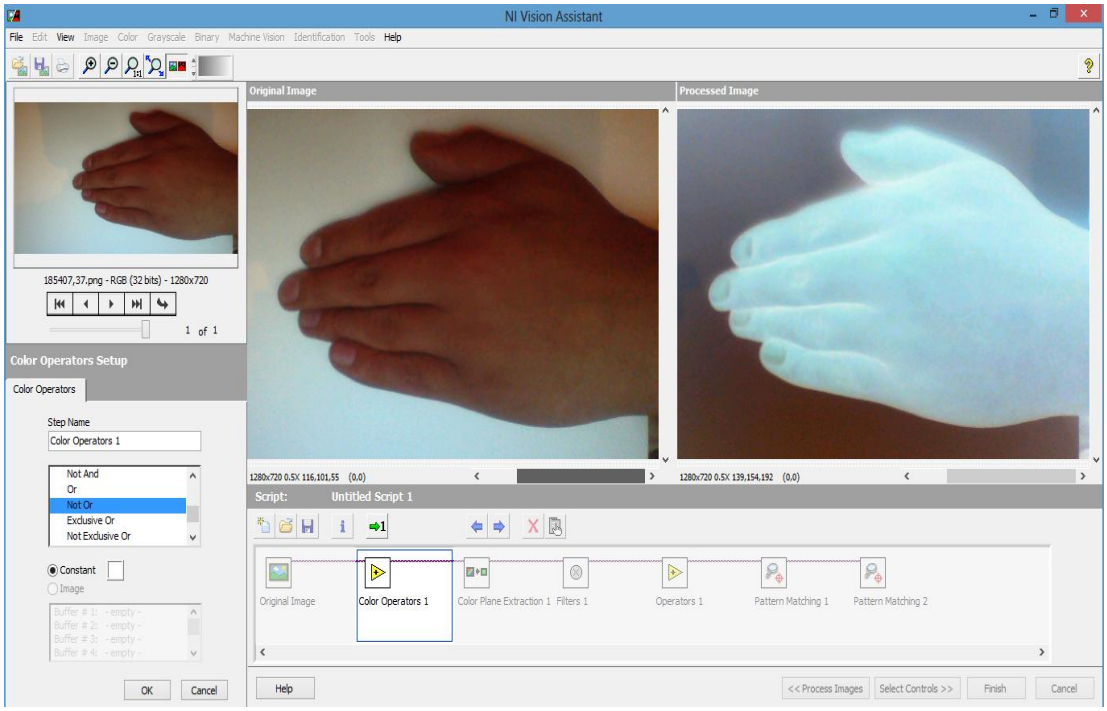


Figura 39. Resultado de aplicar la operación NOT OR al color blanco.

Luego procedemos a transformar la imagen a escala de grises. El plano que se extrae de la imagen RGB es el plano G.

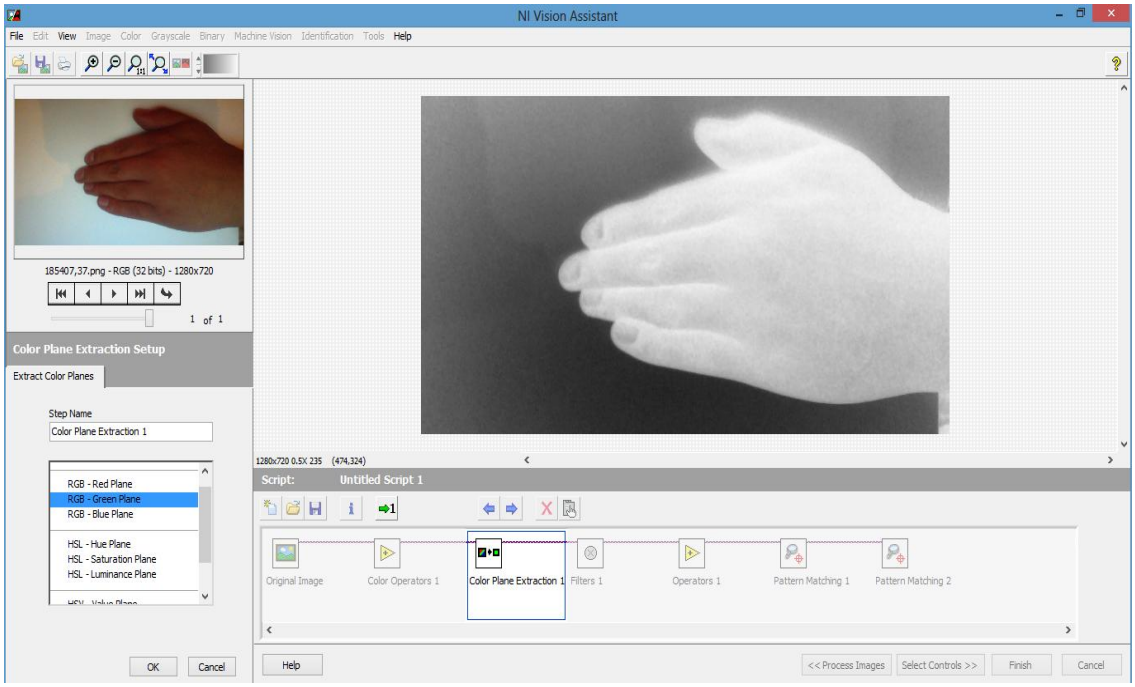


Figura 40. Resultado de extraer el plano G de la Imagen.

En esta parte aplicamos un filtro Gaussiano para eliminar el ruido presente en la imagen.

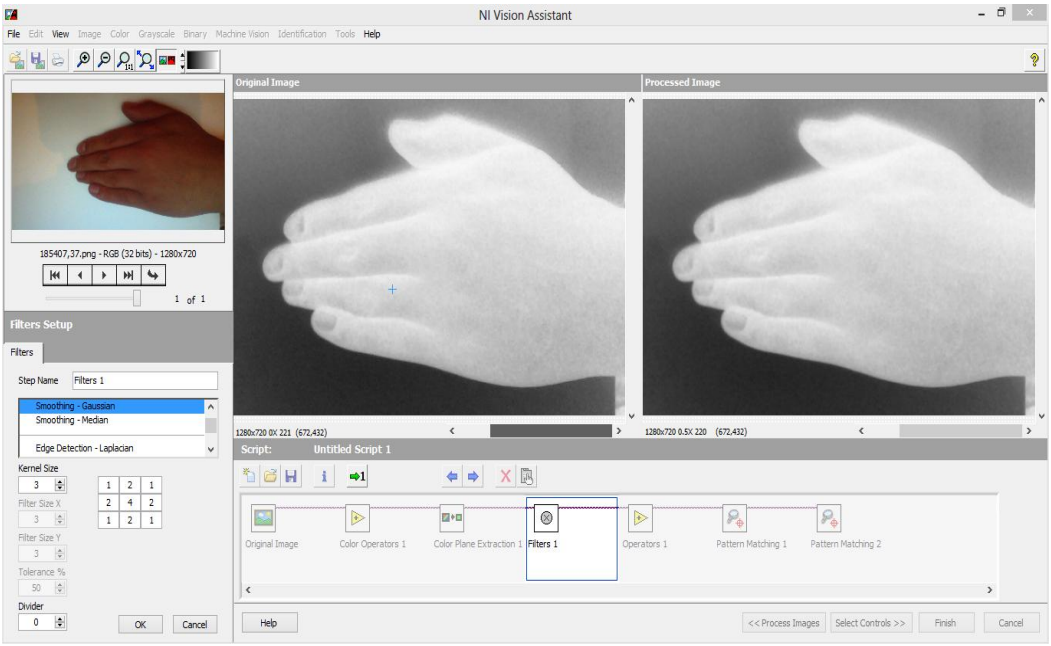


Figura 41. Resultado aplicar el filtro a la imagen.

Posteriormente volvemos a atenuar el color blanco para lo cual restamos un valor dentro de la escala de grises.

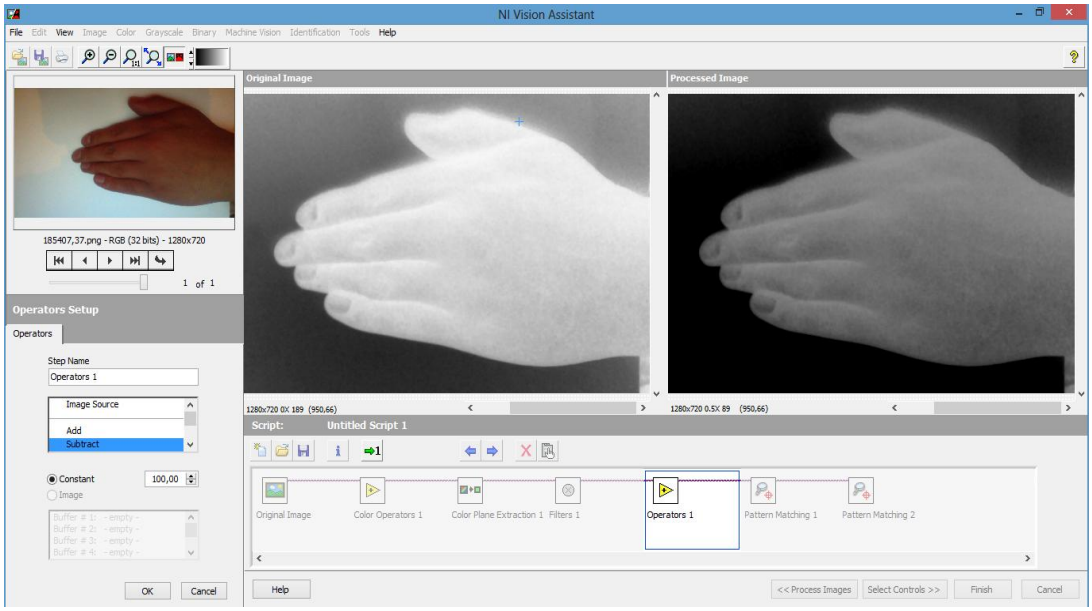


Figura 42. Resultado aplicar la operación resta a la imagen.

Por ultimo procedemos a comparar con los patrones a los cuales se denominan “templates”. En esta parte se hace el reconocimiento propiamente de la señal.

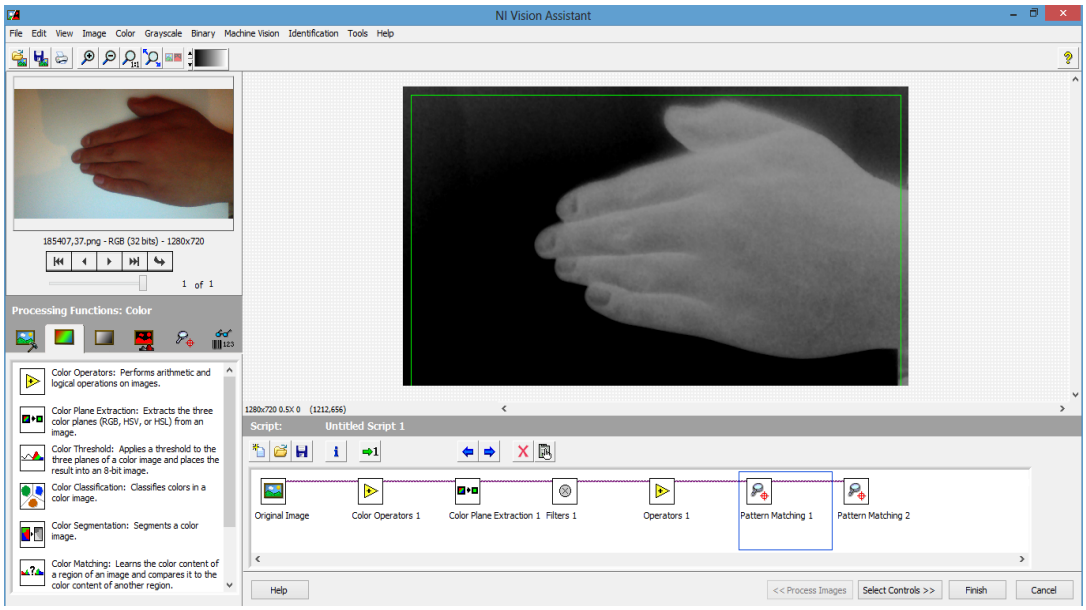


Figura 43. Comparación de la imagen con los patrones.

Una vez que hemos establecido el funcionamiento del software ponemos en finalizar y el asistente nos devuelve el código para ejecutarse en Labview en un bloque como se muestra a continuación.

Figura 44. Código para ejecutarse en Labview.

Algo que resaltar, es que en el código solo empleamos la búsqueda de dos patrones, es decir, solo buscamos dos señales, pero para nuestro control necesitamos 8 señales. La razón es que para cada carga tenemos que crear un bloque diferente. Aunque se podría hacer en un solo bloque, pero para estructurar mejor el programa se decidió hacer un bloque para cada carga.

2.5.1 Desarrollo de La Comunicación Serial

Una vez que realizamos el código para todas las cargas debemos preocuparnos como comunicar el programa con los actuadores externos. Optamos en usar comunicación serial RS 232 y de esta manera enviar datos a un receptor que en este caso es una placa Arduino uno.

Para incluir la comunicación serial a nuestro programa debemos usar las herramientas dentro de la paleta “Intrument I/O” específicamente la herramienta Serial.

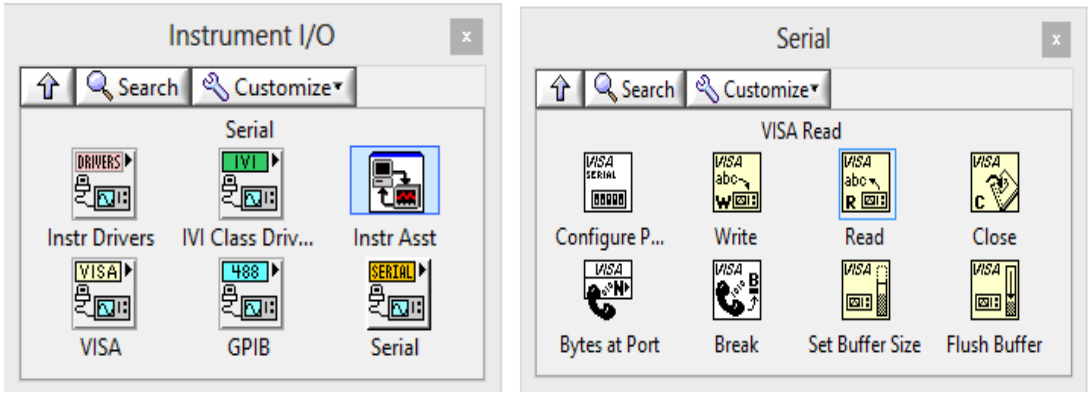


Figura 45. Herramientas para comunicación Serial.

Para nuestro caso el codigo desarrollado para la comunicaion serial es el siguiente:

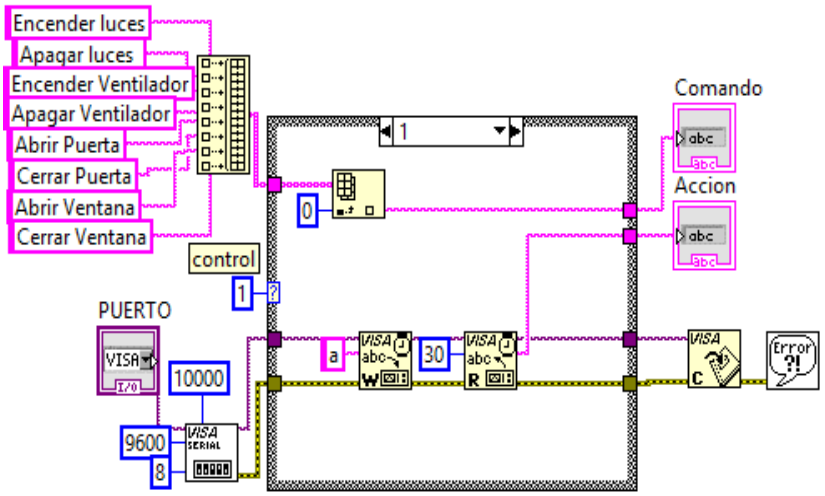


Figura 46. Código de Comunicación Serial.

Debido a que tenemos 8 señales a identificar se creó una estructura caso el cual envié un código que corresponde a una señal como ilustra en la figura anterior. Estos códigos se describen en la siguiente tabla.

CÓDIGO	**ACCION**
a	Encender Luces
b	Apagar Luces
c	Encender Ventilador
d	Apagar Ventilador
e	Abrir Puerta
f	Cerrar Puerta
g	Abrir Ventana
h	Cerrar Ventana

Tabla 1. Descripción de los códigos empleado en la comunicación serial

2.5.2 Desarrollo de Patrones

Los patrones son imágenes de las señales que van a ejecutar el control. Cuando el software adquiere una imagen de la cámara esta se compara con los patrones previamente almacenados y si existe coincidencia o semejanza valorado por una tolerancia, entonces envía un código antes descrito a través de la comunicación serial para que se ejecute la acción correspondiente.

Estos patrones se pueden ver en la siguiente tabla.

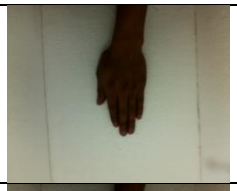
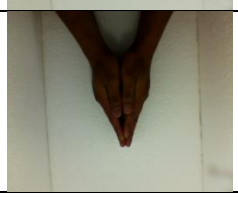
No.	Patrón	Acción
1		Encender luces
2		Apagar luces
3		Encender ventilador
4		Apagar ventilador
5		Abrir puerta
6		Cerrar puerta
7		Abrir ventana
8		Cerrar ventana

Tabla 2. Patrones para el control de cargas.

Juntando todo lo anterior obtenemos la siguiente interfaz usuario que servirá para reconocer las señales de control.

Figura 47. Interfaz usuario del Software.

La interfaz con el usuario está diseñada de manera sencilla para su fácil manejo, consta de una pantalla donde se podrá apreciar las señas que se realicen, se tiene un bloque para controlar la tolerancia para la comparación de la señal realizada en tiempo real con el patrón que previamente ha sido definido, además se tiene dos cuadro de texto que indicaran la acción a realizar y en el otro dará a conocer si la acción fue o no realizada y como adicionales tenemos el botón de stop.

A continuación se muestra una fracción del código utilizado para la interfaz entre el usuario, en el respectivo anexo se podrá ver el código completo.

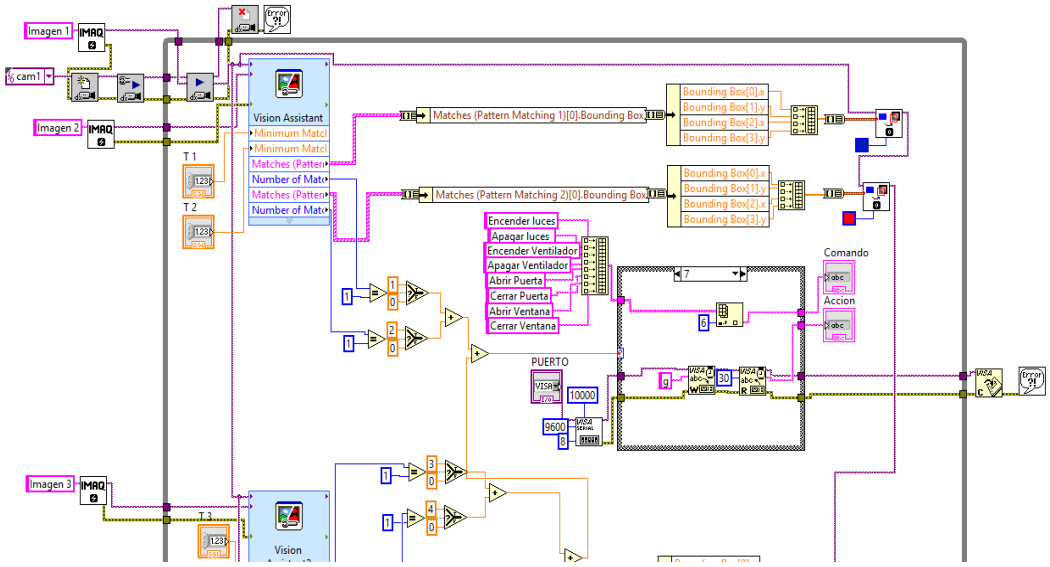


Figura 48. Fracción de código de interfaz usuario.

2.6 DESARROLLO DE APLICACIÓN PARA INFORMACION DE OBSTACULOS

Por la necesidad de mejorar el traslado de forma autónoma de una persona con discapacidad visual, se ha pensado en la realización una aplicación que permita dar información de ciertos obstáculos que puedan interferir en su libre traslado o desplazamiento en el interior de su vivienda, por esta razón el software a través de voz indicara al usuario el obstáculo que se encuentre al frente, permitiendo que el discapacitado pueda actuar y evitar tropezar o golpear con ciertos objetos.

Para la realización de esta aplicación, se ha utilizado los mismos componentes que la aplicación anterior, con la diferencia de que ahora se va a trabajar con colores.

Para la información de obstáculos, se propone implementar una señalización de varios muebles y objetos con franjas de colores para que el respectivo software y con la ayuda de una cámara se pueda clasificar el tipo de obstáculo y dar una notificación.

Se ha optado por identificar colores ya que se tiene mayor efectividad a la hora de comparar con los patrones, ya que no se considera forma ni magnitud del objeto a identificar.

Para ello se ha considerado cinco colores y en la siguiente tabla se da a conocer el color que corresponde a cada obstáculo.

COLOR	OBJETO	INFORMACION
ROJO	PUERTA	Puerta cerrada
AZUL	VENTANA	Ventana cerrada
VERDE	MESA	Mesa al frente
AMARILLO	SILLA	Silla al frente
MORADO	PARED	Pared al frente

Tabla 3. Asignación de colores para cada obstáculo

Para lograr la identificación de cada uno de los colores se utiliza la Labview, en el mismo se ha procedido de la siguiente manera.

2.6.1 Adquisición de la imagen

Para la adquisición de la imagen se utiliza una cámara "'Genius", "FaceCam 1000" y las herramientas de "NI-IMAQdx", para luego realizar el respectivo procesamiento.

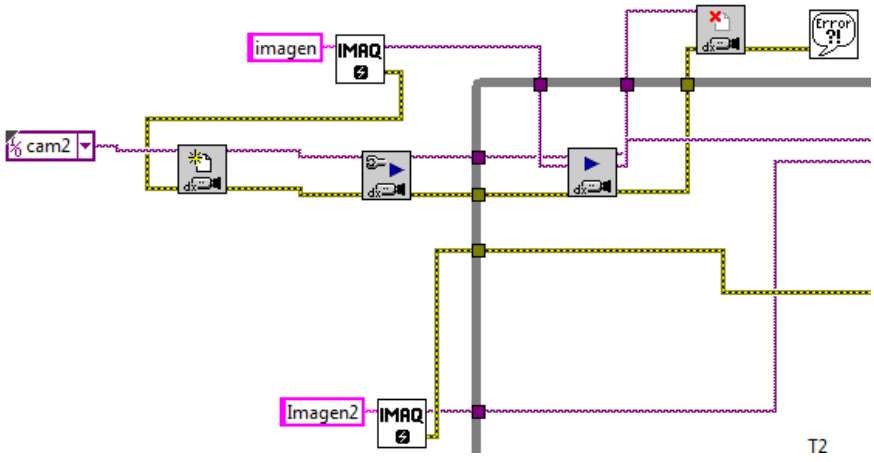


Figura 49. Código para adquisición de imagen.

La imagen que se podrá ver tendrá que ser a color para luego hacer la respectiva comparación con el patrón guardado.

2.6.2 Procesamiento de la imagen y creación de "template"

Para ello se utiliza el asistente de Visión denominado "Vission Assistant", el mismo nos permite crear y procesar los patrones para luego hacer la respectiva comparación, para este caso se ha utilizado solo un bloque, el cual trabajara con los cinco colores que se han planteado inicialmente.

Figura 50. Bloque de "Vision Assistant"

Abriendo el asistente se crea para cada color un "Color Pattern Matching", el cual permite comprobar la presencia de una plantilla "templete" en toda la imagen de color o en una región de interés.

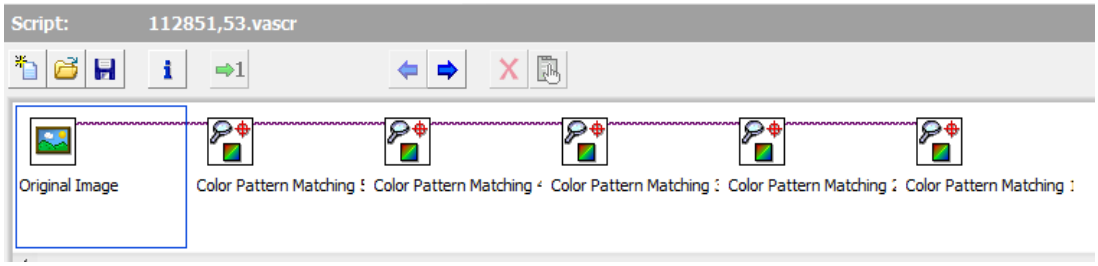


Figura 51. Creacion de "Color Pattern Matching"

El siguiente paso es crear el “template” para que el “Color Pattern Matching” pueda detectarlo cuando la cámara este adquiriendo imágenes en tiempo real y luego poder realizar acciones en base al color identificado.

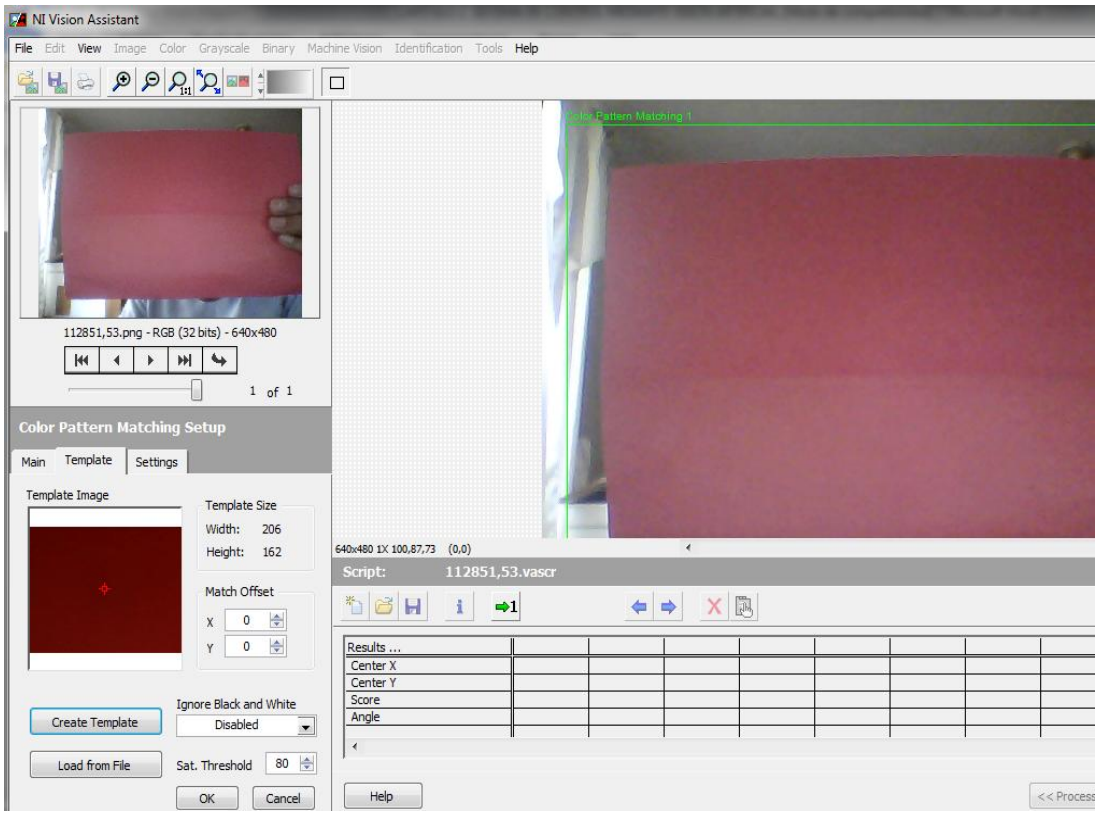


Figura 52. Creacion del “template” para la identificación de puerta cerrada.

2.6.3 Algoritmo para la comparación e identificación de obstáculos

Una vez que “Vision Assistant” realice la respectiva comparación de colores se propone un algoritmo que permita dar a conocer el obstáculo detectado, para ello se utiliza una secuencia de operadores lógicos y una estructura “case”, los que permiten dar a conocer a través de voz el obstáculo detectado.

2.6.4 Interfaz con el usuario

A continuación se muestra la pantalla que permite el interfaz con el usuario, la misma que indica el tipo de obstáculo a través de texto, led a color y voz, dando facilidad al discapacitado tener una información básica para su traslado de un lugar a otro, por ello una vez que la aplicación detecta el

obstáculo, dará la respectiva notificación y el discapacitado podrá tomar una decisión para cambiar su ruta, abrir una puerta o ventana para su respectivo acceso.

Se debe aclarar que la aplicación solamente informa del obstáculo, no da coordenadas de ubicación del objeto ni trayecto a seguir por el usuario, ya que el objetivo es solo saber si hay un obstáculo al frente para que el propio usuario decida que hacer.

Figura 53. Interfaz con el usuario para información de obstáculos.

Para comprobar el correcto funcionamiento se ha propuesto la utilización de paletas con diferentes colores, los cuales simularan a los obstáculos, que para una aplicación real cada uno de los mismos deberán estar señalizados con los colores propuestos anteriormente.

Figura 54. Colores para la señalización de obstáculos

Cada uno de estos colores deben ser colocados en el interior de la caja de adquisición para simular obstáculos al frente de la cámara y la aplicación se encarga de identificarlos.

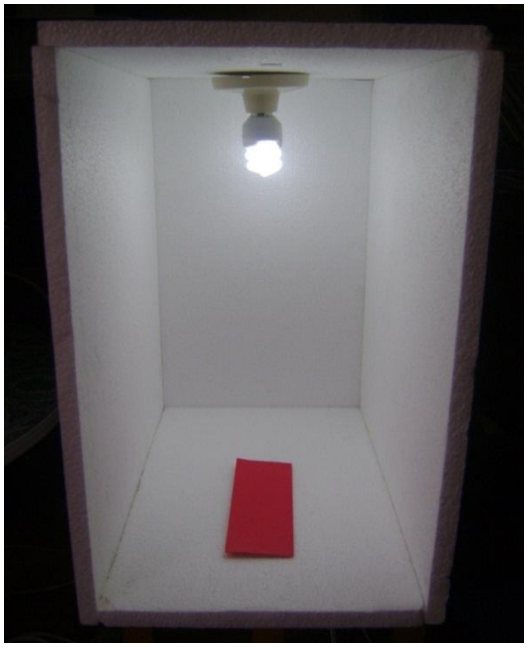


Figura 55. Caja de adquisición

La cámara se encuentra ubica en la parte superior de la caja, por lo tanto los colores se los puede desplazar en cualquier dirección y la aplicación los detecta eficazmente, ya que el ambiente es ideal. Para una aplicación en un ambiente real se deberá hacer algunas mejoras adicionales.

CAPITULO 3

3.1 SISTEMA DE CONTROL MEDIANTE COMANDOS DE VOZ

3.1.1 Sistema domótico contralado mediante comandos de voz

Una vez realizado el análisis correspondiente de las necesidades técnicas se ha optado por la realización de un sistema de control, que nos permita facilitar y dar mayor confort a una persona con cierto grado de discapacidad, por lo cual se determina hacer el control de cargas básicas en una vivienda como son la iluminación, puertas, ventanas y ventiladores, con las cuales el discapacitado podrá actuar y desenvolverse por su cuenta en el caso que desee controlar unas de las cargas antes citadas.

A demás se incluye una alerta de emergencia la cual permitirá a la persona discapacitada realizar un llamado inmediato al personal encargado de su cuidado, logrando que dicho personal no esté en todo momento junto al discapacitado, sino que en caso de necesidad será notificado.

3.2 SOFWARE PARA COMANDOS DE VOZ

Para la realización de la aplicación que interactué con el usuario a través de comandos de voz, se hace necesario el desarrollo de software e implementación de hardware para el sistema de control. Para el caso de reconocimiento de comandos se utiliza Dragon Naturally Speaking y el interfaz con el usuario se lo realiza con Visual Basic 2008 Express.

La aplicación desarrollada en Visual Basic conjuntamente con el software de reconocimiento de voz denominado Dragon Naturally Speaking, el mismo que escribe lo que se dicta, permitirá recibir comandos para que estos realicen alguna tarea previamente definida. Pare ello empezaremos conociendo el software de reconocimiento de voz, para ello seguiremos el siguiente esquema.

- Familiarizarse con los instrumentos que utiliza el programa Dragon Naturally Speaking 10.1

- Desarrollar el entrenamiento de voz para iniciar el trabajo.

- Conocer la lista de comandos que Dragon Naturally Speaking 10.1 proporciona para su fácil manejo.

- Conocer la aplicación en Visual Basic 2010 Express, sus comandos y su modo de utilización.

3.2.1 Dragon Naturally Speaking 10.1

A continuación se desarrolla un análisis del porque se utiliza este software y como usarlo para ello se tiene la siguiente temática: instalación del programa, adaptación al modo de hablar, entrenamiento en dictados y aprendizaje de comandos.

3.2.1.1 Qué es Dragon Naturally Speaking 10.1?

Es un software de reconocimiento de voz que nos permite hablarle a nuestra PC en lugar de escribir. Este nos permite dictar y escribir en Microsoft Word, Excel, Chat, así como a programas de correo electrónico y en casi todos los programas donde se escribe normalmente. Se denomina "dictar" a la acción de hablar a su PC mientras el programa escribe lo que dice.[20]

Para el reconocimiento de voz Dragon trabaja con sofisticados modelos acústicos y estadísticos que le permiten adaptarse rápidamente al usuario de distintas formas, familiarizándose con el sonido de su voz y con las palabras que utiliza al dictar, además aprende de su voz y pronunciación a medida que lo utiliza. Al usar palabras que el programa no conoce, se produce un falso reconocimiento. Corrigiendo estos errores, ayudará al programa a mejorar la capacidad de reconocer su forma de hablar.[21]

[20] NAUNCE. (s.f.). *NAUNCE*. Recuperado el 3 de Mayo de 2013, de www.naunce.com/Dragon
[21] NAUNCE. (s.f.). *NAUNCE*. Recuperado el 3 de Mayo de 2013, de www.naunce.com/Dragon

En el caso de la aplicación desarrollada para reconocimiento de comandos de voz es necesario un entrenamiento básico ya que la aplicación está destinada a realizar acciones de control que le permitan interactuar con la iluminación, motores y alarma de emergencia según los comandos preestablecidos para el efecto.

3.2.1.2 Instalación de Dragon Naturally Speaking 10.1[22]

Antes de instalar el software.- Se debe realizar las siguientes recomendaciones:

- Cierre todas las aplicaciones abiertas.
- Cierre o desactive los programas de antivirus, el proceso de instalación puede activar una falsa alerta de virus.
- Elija el tipo de instalación que desea realizar.

Instalación en una ubicación personalizada.

La carpeta de instalación predeterminada de Dragon es: C:\Archivos de programa\Nuance\NaturallySpeaking10.1, durante el proceso de instalación, tiene la opción de instalar en otra carpeta o disco duro de su equipo. Independientemente de dónde instale los archivos de programa de Dragon, el programa siempre instalará los idiomas y el vocabulario que se seleccionaron durante la instalación en la unidad C, junto con los perfiles de usuarios.

Una vez dada la ubicación de instalación, realice la respectiva instalación del programa en su computador siguiendo las instrucciones y pantallas desarrolladas por la aplicación que guiarán intuitivamente al usuario durante la instalación del programa.

[22] NUANCE COMMUNICATIONS, I. (2011). *Dragon Naturally Speaking, Guia de Usuario, VERSIÓN 11.*

2.2.1.3 Creación de perfiles de usuarios[23]

Una vez instalado el programa debemos crear perfiles de usuario y por ello antes de comenzar a utilizar Dragon, debe crear un perfil de usuario para cada persona que vaya a realizar dictados.

El perfil de usuario contiene información sobre su voz que Dragon utiliza para reconocer lo que dice, por ejemplo, palabras especializadas, nombres, acrónimos y abreviaturas que pueda agregar. La primera vez que inicie el programa, se abrirá el asistente Creación de perfil, el cual le guiará a través del proceso de creación del perfil de usuario.

Figura 56. Pantalla para crear perfil de usuario

Para la creación de un perfil de usuario se debe seguir los siguientes pasos:

1. En la página crear un perfil de usuario, haga clic en el botón Siguiente situado en la esquina inferior derecha para continuar.
2. Siga las indicaciones del asistente, rellenando la información que solicite y haciendo clic en Siguiente para avanzar. Cuando el asistente le solicite un nombre, puede escribir su nombre o un apodo .El

[23] NUANCE COMMUNICATIONS, I. (2011). Dragon Naturally Speaking, Guia de Usuario*, VERSIÓN 11.*

asistente le preguntará su edad para ayudar a Dragon a trabajar con las diferencias vocales que asocia a los distintos rangos de edades.

El asistente también le solicitará que indique la región del mundo donde vive para tener en cuenta las diferencias regionales a la hora de interpretar su voz.

3. En la página ayude a Dragon a identificar las palabras que dice, seleccione su acento dependiendo de la región en donde se encuentre.
4. En la página ¿Cómo le habla a su PC?, seleccione el tipo de micrófono que utiliza en la lista Micrófono.
5. En la página repase las opciones, asegúrese de que la información introducida es correcta antes de continuar.
6. Dragon elegirá automáticamente un modelo de voz y lo utilizará como base para crear su perfil de usuario individual. También seleccionará el vocabulario que utilizará para reconocer las palabras correctamente, basándose tanto en el sonido como en el contexto.

- Modelo de voz: el programa utiliza el modelo de voz para adaptarse a su voz durante el entrenamiento:

  BestMatch III

  BestMatch IV

  Bluetooth 8 kHz

  Bluetooth BestMatch IV

- Vocabulario: puede seleccionar:

  BestMatch Plus

  Sin palabras de dictado

  De forma automática, Dragon recomienda el tipo de modelo de voz y vocabulario más adecuados para la velocidad/memoria de su PC.

7. Haga clic en Aceptar para guardar las opciones seleccionadas en la página elija modelos y volver al asistente Creación de perfil.

8. Haga clic en el botón Crear, situado en la esquina inferior derecha de la página, para crear el perfil de usuario. Cuando termine de crear el perfil de usuario, le llevará a la siguiente fase del proceso.

3.2.1.4 Configuración y colocación del micrófono.[24]

Una vez que el programa haya creado el perfil de usuario, abrirá la página elija el dispositivo que usara para el reconocimiento de voz.

1. La colocación del micrófono es muy importante. Si su posición no es correcta, Dragon no le oirá con claridad y puede cometer más errores.
2. Coloque el micrófono siguiendo las indicaciones del asistente. Haga clic en Siguiente para que el asistente, comprobar micrófono, guíe por el proceso de configuración del micrófono.

También es importante usar siempre la misma posición. Trate de mantener constante la posición del micrófono cada vez que dicte.

Figura 57. Colocación del micrófono

[24] NUANCE COMMUNICATIONS, I. (2011). Dragon Naturally Speaking, Guia de Usuario, *VERSIÓN 11.*

3.2.1.5 Uso de un micrófono auricular[25]

- Coloque el micrófono a menos de dos centímetros de su boca (aproximadamente, la medida de su pulgar) y un poco ladeado. El micrófono no debe tocar la boca, pero casi puede rozar los labios.
- Si tiene que apartar el micrófono, mueva el brazo del micrófono hacia arriba sobre la cabeza, en lugar de flexionarlo para apartarlo o quitarse el auricular.

3.2.1.6 Comprobación del volumen y la calidad de sonido del micrófono[2]

Una vez colocado el micrófono, haga clic en Siguiente y se abrirá el asistente de Dragon para ajustar el volumen para ello realice lo siguiente:

1. Haga clic en el botón Iniciar prueba del volumen y lea en voz alta el texto que aparece en el recuadro.
2. El programa emitirá una señal para indicar que ha finalizado la comprobación del volumen.

3.2.1.7 Entrenamiento de DRAGON NATURALLYSPEAKING 10

Al no existir dos personas que posean la misma voz, se tiene que invertir un poco de tiempo para entrenar a Drago NaturallySpeaking 10.1 para que se adapte a su pronunciación.

Normalmente son cinco minutos para que el software se adapte a su pronunciación. Este entrenamiento debe realizarse la primera vez que inicia el programa; este proceso consiste en la lectura en voz alta de algunos párrafos de una historia o un libro, estos los proporciona el entrenador. Esta información sobre su voz se guardará en un archivo.

El entrenamiento se lo hace siguiendo los siguientes pasos:

[25] NUANCE COMMUNICATIONS, I. (2011). *Dragon Naturally Speaking, Guia de Usuario, VERSIÓN 11.*

1. Lo primero es hacer una prueba de la entrada de audio del sistema de sonido, la relación voz-ruido.

Para eso daremos clic en **"**Iniciar prueba de calidad**"** y nuevamente leeremos en voz alta el texto hasta que el programa nos de la señal de pasar al siguiente paso.

Antes de iniciar la prueba de calidad se nos mostrara la pantalla siguiente que nos permite calibrar el volumen del micrófono.

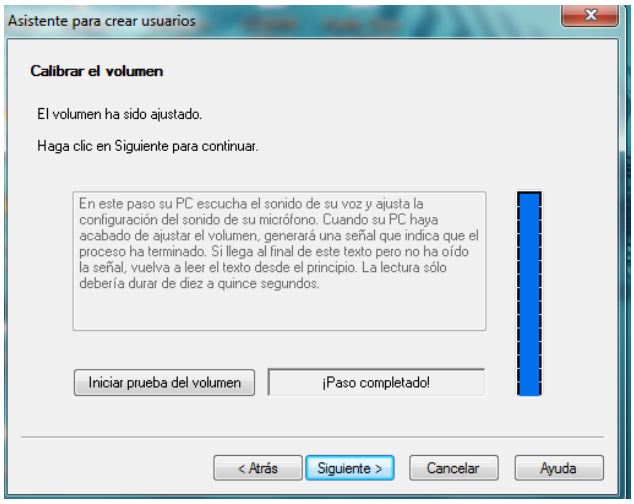


Figura 58. Calibración de volumen del micrófono.

Una vez realizado la calibración de volumen se nos mostrara la siguiente pantalla en donde se realizara la prueba de calidad de la entrada de audio.

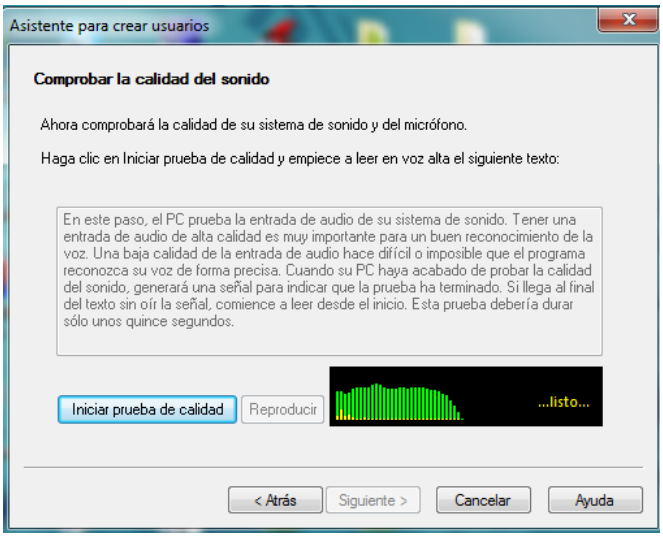


Figura 59. Comprobación de calidad de sonido

2. Elija el texto que leerá en voz alta. Esta lectura le tomará tan solo cinco minutos. En ocasiones, aparecerá un nuevo texto al terminar el elegido. Esto es porque el programa necesita recopilar más información.

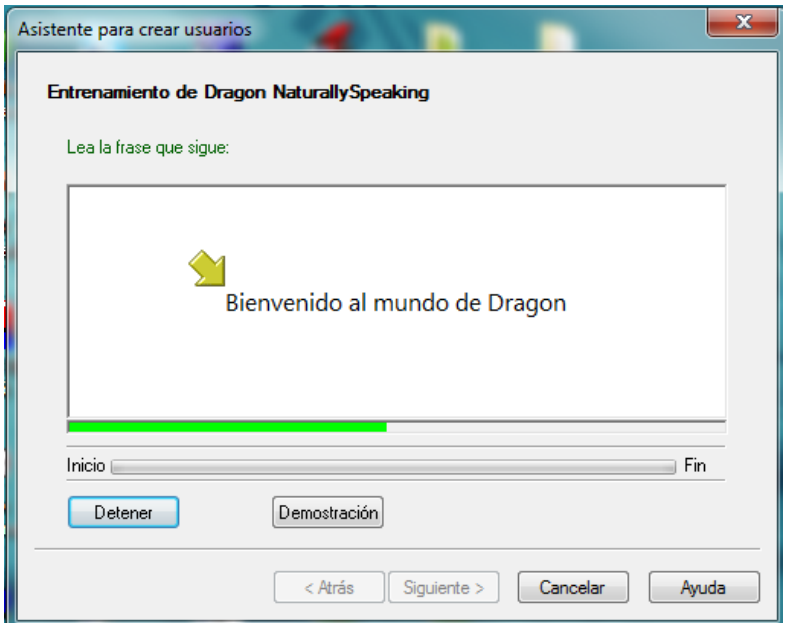


Figura 60. Pantalla para el entrenamiento de Dragon

Una flecha amarilla nos indicará donde comenzará a leer. Para hacer pausa durante el entrenamiento, solo haremos clic en **"**Detener**"**.

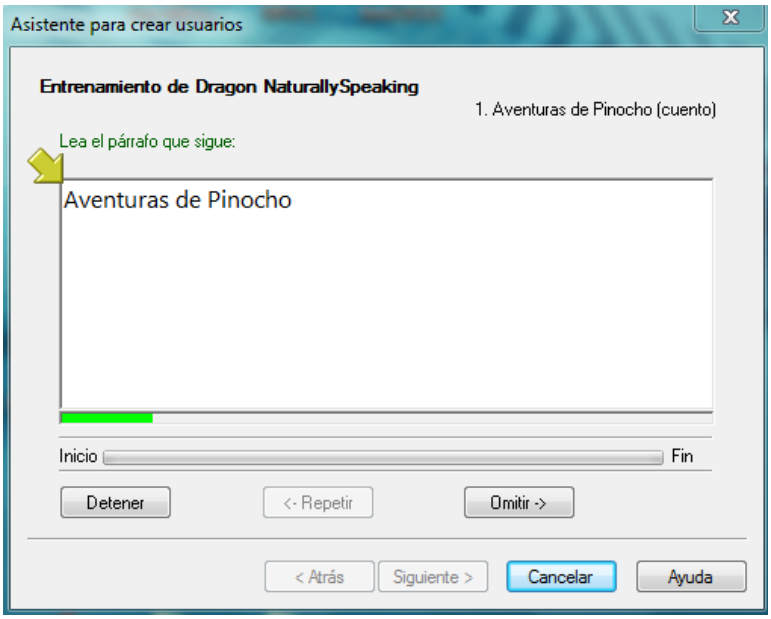


Figura 61. Pantalla para la lectura del texto elegido.

Para pasar las dos primeras pantallas, debe decir las frases sin realizar ninguna pausa. En el momento que haya finalizado el entrenamiento aparecerá la siguiente ventana:

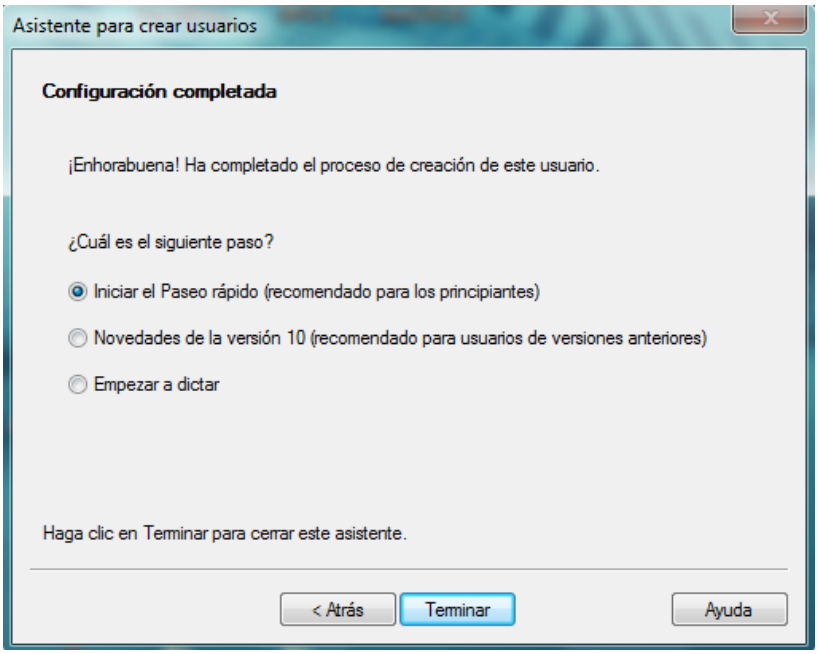


Figura 62. Pantalla de finalización de entrenamiento

En la ventana existen tres opciones, de las cuales elegiremos una antes de finalizar el asistente para crear nuestro usuario.

- Iniciar el tutorial: si es la primera vez que utiliza Dragon.
- Novedades de la versión: si está actualizando una versión anterior.
- Empezar a dictar.

Después de seleccionar la opción, haga clic en terminar.

3.2.1.8 Comenzar a dictar con Dragron Naturally Speaking 10.1[26]

Haga doble clic en el siguiente icono de Dragon en el escritorio del PC.

Figura 63. Icono de Dragon Naturally Speaking 10.1

[26] NUANCE COMMUNICATIONS, I. (2011). *Dragon Naturally Speaking, Guia de Usuario, VERSIÓN 11.*

Inmediatamente se abrirá la barra de “Dragon” que contiene todos los menús los cuales puede utilizar mientras trabaja en Dragon NaturallySpeaking.

Figura 64. Barra de menú de “Dragon”

Antes de dictar, es necesario encender el micrófono. Para ello haga clic en el icono del micrófono en la DragonBar. Puede volver a hacer clic en ese icono para apagarlo.

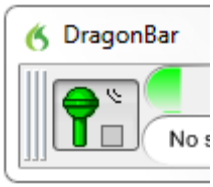


Figura 65. Icono que indica el encendido del micrófono.

Para empezar a dictar, inicie un procesador de textos (como Microsoft® Word) y abra un documento nuevo. Confirme que el cursor esté al principio del nuevo documento.

Para hacer un dictado adecuado, sigan las siguientes instrucciones:

- Hable con claridad y pronuncie todas las letras. No susurré ni hable demasiado fuerte.
- Hable en frases completas.
- Piense lo que va decir antes de hacerlo.

Las primeras palabras que dicte tardan un poco aparecer en la pantalla.

Al empezar a hablar irá apareciendo el texto en el cuadro de resultado mientras Dragon NaturallySpeaking resuelve lo que ha dicho. Este cuadro se trata de una pequeña ventana amarilla que aparece en la pantalla mientras está dictando.

Figura 66. Cuadro de resultados

Algunas veces el programa escribirá algo parecido a lo que ha dicho, pero que no es correcto, por ello es mejor hablar de forma "natural", utilizando frases largas. En caso de que escriba de forma incorrecta lo que dicta se debe hacer su respectiva corrección para que así en la siguiente vez no cometa el mismo error, por último se recalca que mientras más veces se utilice el programa de reconocimiento de voz, se obtendrá mejores resultados y se tendrá cada vez menos correcciones que realizar.

3.2.2 Software para interfaz con el usuario

Para la realización de la interfaz de la aplicación de comandos de voz con el usuario que posee cierto grado de discapacidad motriz se ha optado por la utilización de Visual Basic Express 2010, por ello se dará un breve repaso de las opciones de programación que permite realizar dicho software.

3.2.2.1 Proceso para desarrollar una aplicación en Visual Basic 2010 Express.[27]

Este software facilita el proceso de desarrollar aplicaciones que permitan interactuar con el usuario; en la mayoría de los casos, el proceso consta de los pasos siguientes:

1. **Cree un proyecto.** Un proyecto contiene todos los archivos necesarios para la aplicación y almacena información sobre la aplicación. A veces, una aplicación contendrá más de un proyecto, por ejemplo, un proyecto de aplicación para Windows y uno o varios proyectos de biblioteca de clases.

[27] Gaitan, E. G. (2009). *Paseo por Visual Basic 2008.* Granada, Nicaragua.

2. **Diseñe la interfaz de usuario.** Para ello, puede arrastrar distintos controles, como botones y cuadros de texto, a una superficie de diseño conocida como *formulario*. Puede establecer propiedades que definan el aspecto y comportamiento del formulario y de sus controles.

3. **Escriba el código**. A continuación, tendrá que escribir el código de Visual Basic Express que define cómo se comportará la aplicación y cómo interactuará con el usuario.

4. **Pruebe el código.** Siempre deseará probar la aplicación para asegurarse de que se comporta del modo que esperaba; este proceso se conoce como *depuración.* Visual Basic Express dispone de herramientas de depuración que facilitan la búsqueda y corrección de errores en el código de forma interactiva.

5. **Distribuya la aplicación.** Una vez que la aplicación está completa, puede instalar el programa final en el equipo o distribuirlo y compartirlo con otros usuarios.

3.2.3 Desarrollo de aplicación en Visual Basic 2010 Express

La interfaz gráfica de la aplicación para reconocimiento de comandos de voz, permite al usuario emitir las acciones de control según el conjunto de comandos disponibles para el control de cargas, visualizar el estado de las mismas mediante imágenes y confirmación mediante comandos de voz cuando se haya ejecutado correctamente la acción requerida, permitiendo una gran interactividad entre la aplicación y el usuario discapacitado.

La aplicación para comandos de voz ha sido desarrollada con el objetivo principal de apoyo y ayuda para personas con discapacidad motriz de tal forma que el usuario pueda ejecutar todas las acciones de control a través de comandos de voz sin la necesidad de manipular ningún control adicional.

A continuación se mostrarán las diferentes pantallas y comandos que conforman la aplicación junto con una breve explicación de la programación asignada a cada una de ellas.

3.2.3.1 Pantalla principal

Para la aplicación se ha pensado en un concepto simple, debido a que esta destina a personas con cierto grado de discapacidad. En la pantalla principal se mostrara la respectiva presentación y contenido de la aplicación, se utiliza un “textbox”, en el cual se escribirá los comandos de voz que recibe Dragon Naturally Speaking y partir de los mismos se realizara el control de las cargas los cuales están previamente determinados.

La estructura que se ha planteado es la siguiente:

1. Inicio del programa a través de un botón
2. Dictado del primer comando que permitirá conocer las opciones de control que nos presenta la aplicación
3. Cambio de cuadros de textos de acuerdo a los comandos de voz que reciba la aplicación.
4. Verificación del estado actual de la carga a controlar.
5. Dependiendo del estado de la carga, se debe permitir activar o desactivar la carga.
6. Mostrar el estado actual de la carga mediante texto y voz.
7. Mostrar la acción realizada por el programa mediante texto, imagen y voz.
8. Alarma de alerta, la cual enviara una señal de aviso a la persona que está encargado del cuidado del discapacitado.
9. Realizar una aplicación interactiva con el usuario, que permita tener un fácil manejo de las opciones que presenta dicha aplicación.

A continuación se explica en detalle, que funciones y objetos de Visual Basic Express 2008, se utiliza para obtener la aplicación con las características descritas anteriormente.

Recepción del comando de voz.- Para ello se utiliza un “textbox”, el cual estará todo el tiempo esperando la escritura del comando de voz que recibe Dragon, este es el encargado de dar inicio al algoritmo de control de cargas y envió de la alarma de emergencia las cuales están debidamente programadas.

Figura 67. Pantalla principal de la aplicación

Una vez que el comando sea escrito en el “textbox”, la aplicación mediante archivos de voz dará a conocer las opciones que tiene la misma, estas son control de iluminación, puerta, ventana, persiana, ventilador y envió de alarma de ayuda.

Figura 68. **Pantalla para la elección de la carga a controlar**

3.2.3.2 Verificación de estado actual de la carga.- Una vez que se elige la carga a controlar, se tiene una opción que nos permite verificar el estado actual de la carga, permitiendo así tener claro que acción se deba ejecutar, ya sea la activación o desactivación de cargas.

3.2.3.3 Activación o desactivación de cargas.- Para ello se dictara el comando de voz respectivo el cual dará la señal para que el algoritmo haga todos los pasos necesarios y a través de la comunicación serial con el microcontrolador, se pueda cambiar el estado de la carga, para ello se tiene el respectivo circuito de potencia para el manejo de cada uno de las cargas.

A demás se tiene la confirmación de la ejecución de los comandos dictados, a través de texto y gráficos, para ello se utiliza un “textbox”, un “picture box”. De igual manera mediante archivos de audio.wam se notificara la acción ejecutada.

A continuación se mostrara la pantalla que se podrá observar mientras se ejecuta la aplicación.

Figura 69. Luminaria encendida

Figura 70. Luminaria apagada

Figura 71. Puerta abierta

Figura 72. Puerta cerrada

Figura 73. Ventana abierta

Figura 74. Ventana cerrada

Figura 75. Ventilador activado

Figura 76. Ventilador desactivado

Figura 77. Alarma de ayuda

3.2.3.4 Alarma de emergencia.- Al ser el discapacitado una persona que necesita atención por parte de algún familiar o encargado especial, es necesario optar por un sistema que nos permita dar una alerta, en el momento que necesite ayuda, por ello se tiene un comando de voz que permite realizar dicha acción, con ello se logra que la persona encargada del cuidado no esté constantemente junto al discapacitado, sino más bien se acerque cuando sea necesario.

3.2.4 Comandos de voz para la ejecución la aplicación.

Se ha tratado que los comandos de voz sean lo más simple posible, y en cantidad mínima necesaria, a continuación se dará a conocer el comando y la acción que la misma ejecutara cuando sea dictado en la aplicación.

No.	Comando	Acción a ejecutar
	Inicio	Arranca la aplicación
1	Dictar	Habilita para le dictado
2	Opciones	Coloca la pantalla de elección de cargas
3	Estado actual	Verifica el estado actual de la carga

		seleccionada
4	Encender iluminación	Enciende luminaria
5	Apagar iluminación	Apaga luminaria
6	Abrir puerta	Abre la puerta
7	Cerrar puerta	Cierra la puerta
8	Abrir ventana	Abre la ventana
9	Cerrar ventana	Cierra la ventana
	Activar ventilador	
10	Desactivar ventilador	Desactiva el ventilador
11	Emergencia	Activa alarma de ayuda
12	Salir	Cierra la aplicación

Tabla 4. Comandos de voz que reconoce la aplicación

3.2.5 Algoritmo para la aplicación

En Visual Basic 20010 Express se desarrolla el algoritmo que nos permitirá interactuar con la aplicación, controlar cargas y enviar alarma de ayuda. Al ser amplia la programación, se opta por explicar el control de la iluminación ya que para el resto de cargas tiene la misma estructura simplemente se realiza cambio de variables y algunos ajustes básicos. En uno de los anexo se colocara la programación completa de la aplicación.

Para la interfaz con el hardware se utiliza la comunicación serial, con lo que permite en todo momento saber el estado y actuación de las cargas de acuerdo a los comandos que se dicten.

A continuación se explica la estructura del programa.

Código para control de la iluminación	Descripción
`Private Sub Txt_comando_TextChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Txt_comando.TextChanged`	Evento que se ejecuta cuando Dragon escriba en el "textbox".
`Coman1 = Me.Txt_comando.Text`	Se asigna a la variable Comma1 el texto escrito
`If Coman1 = "iluminación" Then`	Compara el contenido de Coman1 con la palabra iluminación
`IBAN = "i"`	Asignamos a la variable IBAN la letra i
`Me.Coman1 = ""`	Borra el contenido de la variable
`Me.Txt_comando.Text = ""`	Borra el texto escrito en el "text box"
`My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos grabados\7.wav", _ AudioPlayMode.Background)`	Reproduce el archivo de sonido "Verifique estado actual de luminaria "
`End If`	
`If IBAN = "i" Then`	Compara el contenido de la variable IBAN, si es correcto entra en el bucle para el control de la iluminación
`If Coman1 = "estado actual" Then`	Compara el texto que contenga "Coman1"
`SerialPort1.Write("i")`	Envía a través del puerto

	serial la letra i
`If Ivar1 = "encendido" Then`	Compara la variable Ivar1, cuyo contendido cambia según el estado de la carga
`Me.Txt_Imens.Text = "DICTE COMANDO PARA APAGAR LUMINARIA"`	Escribe en el cuadro de mensajes
`Ivar1 = ""`	Limpia variable
`Me.Coman1 = ""`	Limpia variable
`Me.Txt_comando.Text = ""`	Limpia “textbox”
`My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos grabados\21.wav", _ AudioPlayMode.Background)`	Reproduce audio “Dicte comando para apagar luminaria”
`Else`	
`Me.Txt_Imens.Text = "DICTE COMANDO PARA ENCENDER LUMINARIA"`	Escribe en el cuadro de mensajes
`Me.Coman1 = ""`	Limpia variable
`Me.Txt_comando.Text = ""`	Limpia “Textbox”
`My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos grabados\20.wav", _ AudioPlayMode.Background)`	Reproduce audio “Dicte comando para encender luminaria”
`End If`	
`End If`	
`If Coman1 = "apagar luminaria" Then`	Compara el contenido de la variable “Coman1”
`Me.Txt_Imens.Text = "LUCES APAGADAS"`	Escribe en el cuadro de mensajes
`Me.Coman1 = ""`	Limpia variable
`Me.Txt_comando.Text = ""`	Limpia” textbox”

`Try`	

`Ptb_car.Image = New Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\focoapagado.jpg")`
`Catch ex As Exception`
`MessageBox.Show(ex.Message, ex.Source)`
	Carga imagen indicativa de luminaria apagada

`Ptb_car.Image = Nothing`	
`End Try`	
`My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos grabados\10.wav", _` `AudioPlayMode.Background)`	Reproduce audio “Luces apagadas”
`SerialPort1.Write("a")`	Envía dato al Puerto serial
`End If`	
`If Coman1 = "encender luminaria" Then`	Compara el contenido de la variable “Coman1”
`Me.Txt_Imens.Text = "LUCES ENCENDIDAS"`	Escribe en el cuadro de mensajes
`Me.Coman1 = ""`	Limpia variable
`Me.Txt_comando.Text = ""`	Limpia “textbox”
`Try`	

`Ptb_car.Image = New Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\focoencendido.jpg")`
`Catch ex As Exception`
`MessageBox.Show(ex.Message, ex.Source)`
	Carga imagen indicativa de luminaria encendida

`Ptb_car.Image = Nothing`	
`End Try`	

`My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos grabados\9.wav", _`
`AudioPlayMode.Background)`
`SerialPort1.Write("e")`
	Reproduce audio “Luces encendidas”

`End If`	
`If Coman1 = "opciones" Then`	Compara la variable “Coman1”
`IBAN = "s"`	Si IBAN es igual a s, la aplicación se coloca en la pantalla de opciones de carga
`Me.Coman1 = ""`	Limpia

	variable
`Me.Txt_comando.Text = ""`	Limpia "TextBox"
`Me.Txt_Imens.Text = ""`	Limpia cuadro de mensajes
`Try`	

`Ptb_car.Image = New Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\logo.png")`
`Catch ex As Exception`
`MessageBox.Show(ex.Message, ex.Source)`
	Carga logo de la Universidad

`Ptb_car.Image = Nothing`	
`End Try`	
`End If`	
`End If`	
`End sub`	

Tabla 5. Secuencia de algoritmo para el control de la iluminación

La variable "coman1" es una variable tipo "string" a la cual se le asigna el texto que estribe Dragon en el "textbox" destinado a recibir la escritura de los comandos de vos, por ello en la mayoría de las ocasiones se debe comparar dicha variable para que el algoritmo de la aplicación realice su tarea correctamente.

3.3 HARDWARE

Para el desarrollo del prototipo se optado por utilizar un microcontrolador 16F877A, el cual se encargará de recibir los datos a través de la comunicación serial y de igual manera enviara datos hacia la aplicación confirmado las acciones realizadas.

En el capítulo correspondiente al diseño del prototipo se explicara con detalle la construcción del hardware y la construcción respectiva de las tarjetas que se encargaran del control de las cargas de la vivienda domótica.

3.4 GUIA PARA USO DE LA APLICACIÓN DE COMANDOS DE VOZ

Una vez que se tenga instalado la aplicación para control domótico mediante comandos de voz y el software de reconocimiento de voz "Dragon

Naturally Speaking 10.1”, se podrá a comenzar a controlar cargas de una manera sencilla y eficaz, con lo cual la persona discapacitada que posee cierto grado de movilidad, tendrá que estar junto al ordenador y colocarse los auriculares y tendrá la oportunidad de manipular la aplicación según sea su necesidad.

Para el fácil manejo de la aplicación se desarrolla a continuación una pequeña guía de usuario para que el discapacitado se familiarice rápidamente con la aplicación.

1. Ejecute la aplicación de comandos de voz

2. Ejecute Dragon y haga click en icono de activación del micrófono

3. Damos clic en la pantalla principal de la aplicación para que Dragon pueda escribir los comandos de voz sobre la aplicación.

4. Dar clic sobre el botón INICIO para que la aplicación esté lista para recibir los comandos.

5. Dictamos el comando de voz “dictar”, este permitirá conocer las cargas que se pueden controlar.

6. Elija la carga a controlar, para ello dicte uno de los siguientes comandos de voz “iluminación”, “puerta”, “ventana”, “ventilador”

7. Verifique estado actual de la carga con el comando “estado actual”

8. Conocido el estado actual, dicte uno de los siguientes comandos, por ejemplo: “encender iluminación”, “apagar iluminación”, “abrir puerta ”, “cerrar puerta”, “activar ventilador”, “desactivar ventilador; dependido de la carga que este controlando.

9. Para cambiar el tipo de carga a controlar, dicte el comando “opciones” el cual le colocara en la pantalla que permite hacer la elección de la carga

10. Para solicitar ayuda simplemente dicte el comando “emergencia” y este activara una alarma sonora que permita alertar a al individuo que está encargado del cuidado y atención del discapacitado.

11. Para terminar la aplicación dicte “salir”

Si el usuario sigue correctamente la secuencia explicada anteriormente, será muy fácil el manejo de la aplicación, además se debe tener en cuenta que la aplicación por si misma irá guiando mediante texto, gráficos y voz la acción que deber realizar la persona que este manipulando dicha aplicación.

CAPITULO 4

APLICACIÓN EN ANDROID

4.1 CONCEPTOS BÁSICOS

4.1.1 Entorno de Desarrollo de Android

Para empezar a crear aplicaciones en Android necesitamos del entorno de desarrollo que está disponible en su página web. Desde el siguiente enlace:

http://developer.android.com/sdk/index.html#win-bundle

En este entorno viene incluido todas las herramientas para empezar a programar aplicaciones en esta plataforma. Cabe mencionar que para programar en Android se requiere conocimientos básicos de Java y programación orientada a objetos.

Dentro del paquete de desarrollo viene incluido un IDE (Eclipse) el kit de desarrollo de Android (SDK) y el Plugin para Eclipse ADT (Android Developer Tools) que ya viene instalado en el IDE.

Además de su página web existe un sin número de ejemplos y documentación oficial que son de gran ayuda cuando se quiere empezar a programar.

4.1.1.1 Perspectiva Java

Como ya mencionamos Android usa Eclipse que es un IDE conocido por quienes han programado en Java y tiene la siguiente perspectiva.

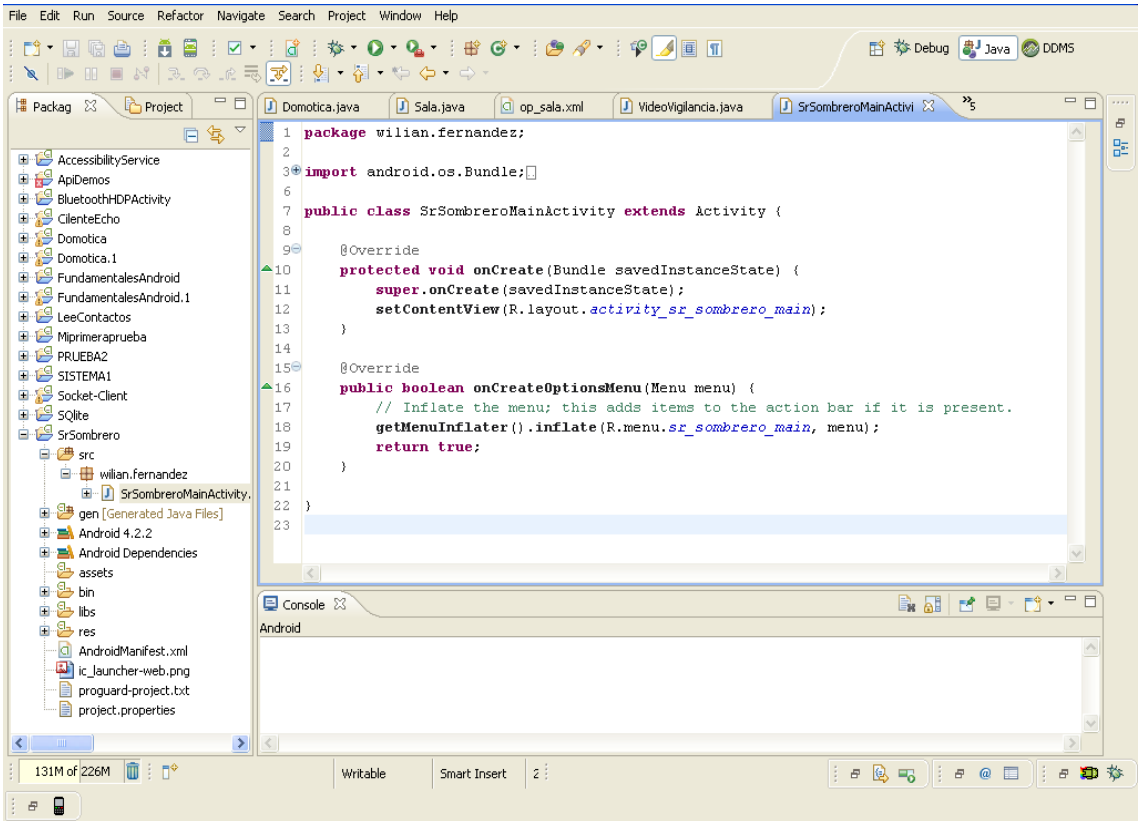


Figura 78. Perspectiva java

Esta es la interfaz usuario que utilizaremos para crear aplicaciones en Android. Esta interfaz, proporciona una serie de herramientas que son de gran ayuda al momento de programar.

4.1.1.2 Perspectiva DDMS

El Plugin “ADT” nos proporciona una nueva perspectiva que nos sirve para corregir errores, interactuar con emulador integrado en el “SDK”, ver hilos de programación, depurar entre otras cosas.

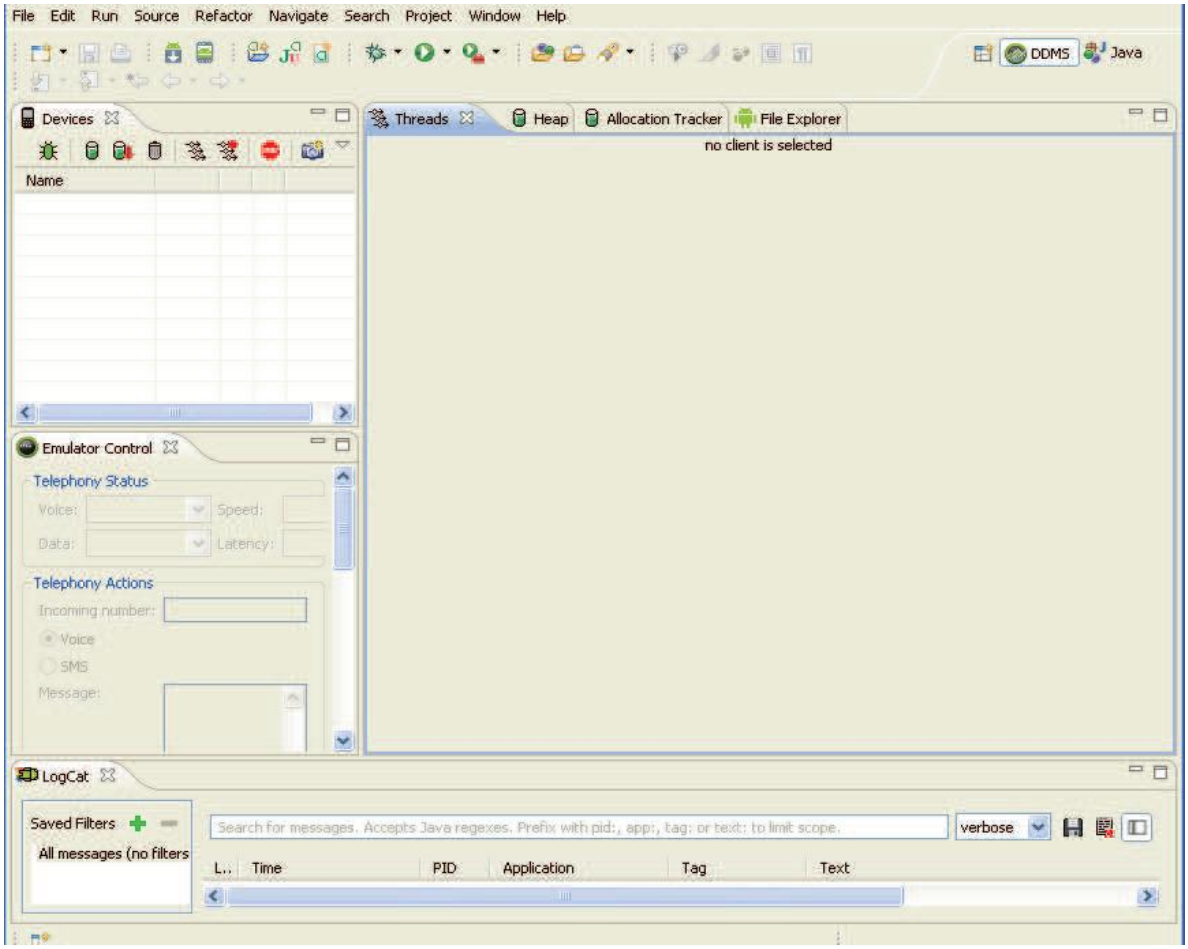


Figura 79. Perspectiva DDMS

4.1.1.3 Emulador

A la hora de probar y depurar aplicaciones no tenemos que hacerlo un dispositivo real. El Kit de desarrollo “SDK” cuenta con un emulador integrado en donde podemos instalar nuestras aplicaciones para ver su funcionamiento. A continuación se muestra la vista del emulador.

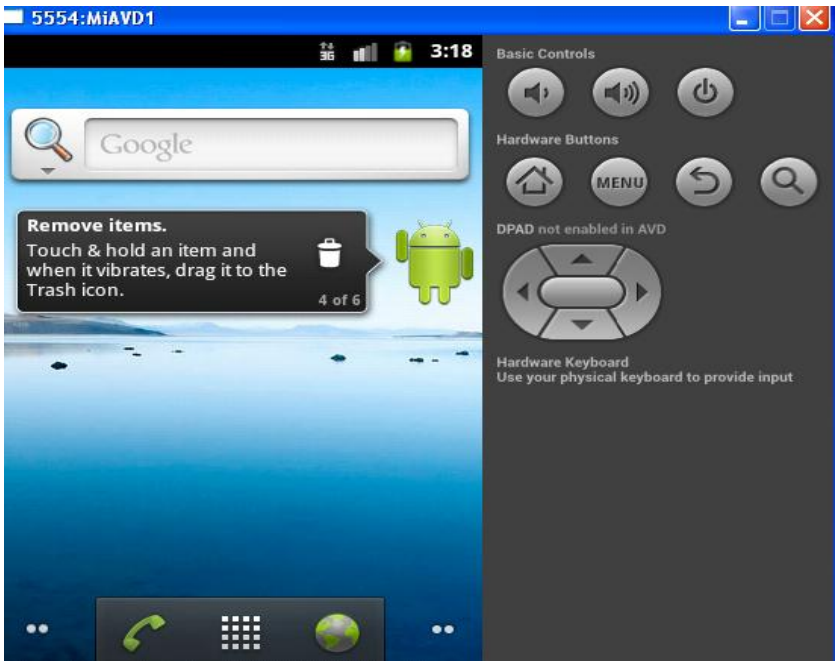


Figura 80. Emulador para Android 2.3.3

No se debe parar la ejecución de un emulador, ya que este cuando se inicia necesita muchos recursos de la computadora, por lo que tarda bastante en ejecutarse, y no es necesario cerrarlo, puesto que cada vez que se lleva a cabo una ejecución del proyecto la aplicación se reinstala en el emulador.

4.1.2 Estructura de un Proyecto Android[28]

Una vez expuesto lo anterior pasamos a comprender cómo se construye una aplicación en Android.

Cuando creamos un nuevo proyecto Android en Eclipse se genera automáticamente una serie de carpetas con la estructura necesaria para poder generar posteriormente la aplicación. Esta estructura será común independientemente del tipo o tamaño de la aplicación

En la siguiente imagen podemos ver la estructura de estas carpetas:

[28] Salvador Gómez Oliver. **"***Curso de Programación Android***"**, Madrid, 2011.

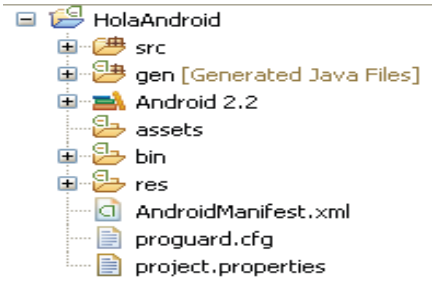
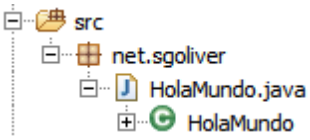


Figura 81. Estructura de las carpetas de un proyecto Android

A continuación veremos el significado de cada una de las carpetas:

4.1.2.1 Carpeta scr

Contiene el código fuente (Java) de la aplicación, clases auxiliares, interfaces, etc. Cuando generamos un proyecto, Eclipse creará automáticamente el código de la ventana principal (Activity).



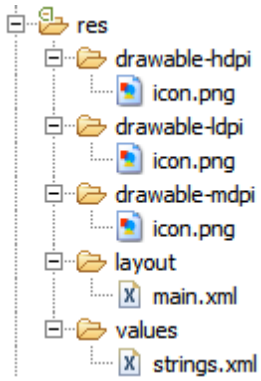
4.1.2.2 Carpeta res

Contiene todos los recursos necesarios por la aplicación: imágenes, videos, cadenas de texto, archivos de audio, etc. Estos recursos están distribuidos en las siguientes carpetas

- **Carpeta drawable:** contiene todas las imágenes de la aplicación y dependiendo de la resolución del dispositivo estarán organizadas en subcarpetas de las cuales las más usuales son: drawable_hdpi, drawable_ldpi, drawable_mdpi.
- **Carpeta layout:** contiene los ficheros de las diferentes pantallas de la interfaz gráfica.
- **Carpeta menu:** contiene los ficheros de los menús de la aplicación.
- **Carpeta values:** contiene otros recursos de la aplicación como por ejemplo: cadenas de texto, estilos, colores, dimensiones, etc. Todos en formato XML.

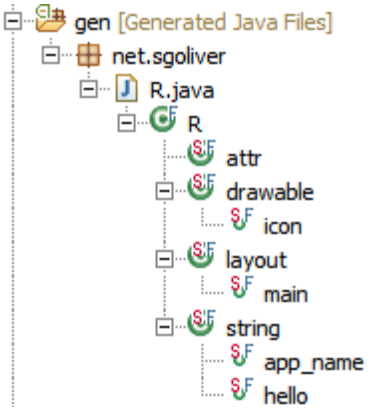
- **Carpeta xml:** contiene todos los archivos XML usados por aplicación.
- **Carpeta raw:** contiene los recursos adicionales, normalmente en un formato distinto a XML, que no se incluye en el resto de carpetas de recursos.

Generalmente para una aplicación se usan los siguientes recursos:



4.1.2.3 Carpeta gen

Esta carpeta guarda un conjunto de archivos (código Java) que se generan automáticamente compilar un proyecto, para poder dirigir los recursos al proyecto. Dentro de esta se genera el archivo R.java y la clase R que contendrá en todo momento una serie de constantes con los identificadores de todos los recursos de la Aplicación incluidos en la carpeta “res”, de tal forma que podamos acceder fácilmente estos recursos mediante esta clase.



4.1.2.4 Carpeta assets

Contiene todos los demás ficheros auxiliares necesarios para la aplicación (y que se incluirán en su propio paquete), como por ejemplo ficheros de configuración, de datos, etc.

La diferencia principal entre estos archivos y los que se guardan en la carpeta “res “es para los primeros no se generan identificadores (ID) como los que genera la clase R mencionada anteriormente. Sin embargo, se puede acceder a los archivos a carpeta “asset” por su ruta usando una clase llamada “AssetsManager”.

4.1.2.5 Archivo AndroidManifest.xml

Es uno de los archivos más importantes de cualquier aplicación Android. Se genera automáticamente a crear el proyecto y en él se encuentra definido los aspectos más importantes de la aplicación, por ejemplo: su identificación (nombre, versión, icono, etc.), sus componentes (pantallas, mensajes, etc.), o permisos para su funcionamiento.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      package="com.wilian.cilenteecho"
4      android:versionCode="1"
5      android:versionName="1.0" >
6
7      <uses-sdk
8          android:minSdkVersion="8"
9          android:targetSdkVersion="17" />
10     <uses-permission android:name="android.permission.INTERNET"/>
11
12     <application
13         android:allowBackup="true"
14         android:icon="@drawable/ic_launcher"
15         android:label="@string/app_name"
16         android:theme="@style/AppTheme" >
17         <activity
18             android:name="com.wilian.cilenteecho.CilenteEcho"
19             android:label="@string/app_name" >
20             <intent-filter>
21                 <action android:name="android.intent.action.MAIN" />
22
23                 <category android:name="android.intent.category.LAUNCHER" />
24             </intent-filter>
25         </activity>
26     </application>
27
28 </manifest>
```

Figura 82. Ejemplo de un archivo AndroidManifest.xml

4.1.3 Componentes de una Aplicación

Anteriormente vimos cómo se estructura un proyecto y vimos dónde colocar cada uno de los elementos que conforman una aplicación. Pero ha llegado el momento de centrarnos en los componentes de tipo software con la que podemos construir una aplicación.

Uno de los aspectos a resaltar es que Android trabaja en Linux, y cada aplicación utiliza un proceso propio. Se distinguen por un ID para que sólo ella tenga acceso a sus archivos. Además, los dispositivos tienen un único foco, es decir, que solo una aplicación está visible en la pantalla, pero puede tener varias aplicaciones en segundo plano, cada una en su propia pila de tareas. Cada pila de tareas se componen a su vez de actividades se van apilando según sean invocadas, y solo pueden terminarse cuando las actividades superiores se han terminado antes, o cuando el sistema las ha eliminado porque necesita memoria. El sistema siempre eliminara la actividad que lleve más tiempo detenida. Si el sistema necesita mucha memoria las aplicaciones en segundo plano pueden ser eliminadas a excepción de su actividad principal.[29]

Figura 83. Pila de tareas o Actividades en Android.

[29] Manuel Báez, Álvaro Borrego, Jorge Cordero, Luis Cruz, Miguel González, Francisco Hernández, David Palomero, José Rodríguez de Llera, Daniel Sanz, Mariam Saucedo, Pilar Torralbo, Álvaro Zapata. *"Introducción a Android",* Madrid, 2011.

Dicho esto, procedemos a indicar los componentes básicos de una aplicación.

4.1.3.1 Actividades (Activity) [30]

Una actividad es el componente principal encargado de mostrar al usuario la interfaz gráfica, es decir, una actividad es un elemento análogo a una ventana en otros lenguajes programación.

Las actividades tienen un ciclo de vida, es decir, pasan por diferentes estados desde que se inician hasta que se destruye, estos estados son:

- **Activo:** ocurre cuando la actividad está en ejecución, es decir, cuando es la tarea principal la cual vemos en la pantalla del dispositivo y podemos interactuar con ella.

- **Pausado:** es cuando la cuando la actividad está en ejecución y visible pero parcialmente obscurecida (semi-suspendida) ya sea por la aparición de un dialogo, otra actividad, etc. Se debe guardar información en este estado para prevenir la posible pérdida de información en el caso de que el sistema decida destruirla para liberar memoria.

- **Detenido:** la actividad no es visible al usuario y el sistema puede destruirla liberar memoria. En el caso de necesitarla de nuevo, se reiniciará desde el principio.

Cabe mencionar que mientras la actividad no se destruya ésta puede regresar a su estado activo de cualquiera de sus otros estados.

[30] Manuel Báez, Álvaro Borrego, Jorge Cordero, Luis Cruz, Miguel González, Francisco Hernández, David Palomero, José Rodríguez de Llera, Daniel Sanz, Mariam Saucedo, Pilar Torralbo, Álvaro Zapata. *"Introducción a Android",* Madrid, 2011.

Además, las actividades cuentan con varios métodos que nos permite saber el cambio de estado que ha tenido en actividad. Estos métodos son las siguientes:

- **OnCreate (Bundle savedInstanceState):** este método es llamado cuando la actividad es iniciada la primera vez y es aquí donde configuramos todos los componentes necesarios para la actividad. Además, este método permite a la Actividad guardar la información proveniente de un estado anterior en su ciclo de vida.

- **Onstart():** este método es llamado inmediatamente después de OnCreate () o después del método OnRestart () proveniente desde un estado detenido cuando la actividad es reiniciada.

- **OnResume():** este método es llamado después del método Onstart() o cuando la actividad es recomenzada desde un estado pause. Establece el inicio de la interactividad entre el usuario y la aplicación. Solo se ejecuta cuando la actividad está en primer plano.

- **OnPause():** se ejecuta cuando una actividad va a dejar de estar en primer plano, para dar paso a otra es decir, es llamado cuando la actividad va entrar en un estado de pausa.

- **OnStop():** este método es llamado cuando la actividad entra en el estado detenido es decir, la actividad pasa a segundo plano por un largo periodo. Va siempre precedido del método OnPause() esto es muy importante porque nos indica que una actividad antes de estar detenida debe estar pausada.

- **OnRestart():** es llamada cuando la actividad es reiniciada desde un estado detenido. Va siempre precedido del método OnStop()

- **OnDestroy():** este método llamado el final del ciclo de la vida de una actividad cuando ésta ya no es necesaria y es destruida de modo irreversible.

Además de estos métodos, cabe destacar dos más, que son de vital importancia aunque no son parte del ciclo de vida de una actividad:

- **OnSavedInstanceState():** guarda el estado de una actividad. Es muy útil cuando se va a pausar una actividad para abrir otra.

- **OnRestoreInstanceState():** restaura los datos guardados en onSavedInstanceState() al reiniciar una actividad.

A continuación se indica un gráfico en donde se encuentra el ciclo de vida de una actividad.

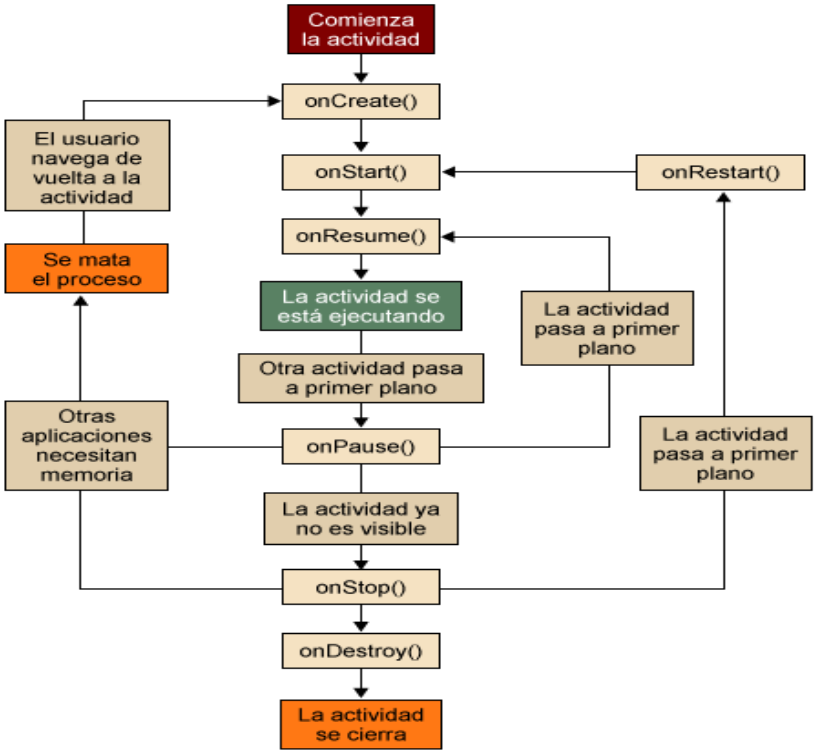


Figura 84. Ciclo de vida de una Actividad.

4.1.3.2 View[31]

Los objetos “view” son los componentes básicos con los que se construye la interfaz gráfica de la aplicación, análogo por ejemplo a los controles de Java .De inicio, Android pone a nuestra disposición una gran cantidad de controles básicos, como cuadros de texto, botones, listas desplegables o imágenes, aunque también existe la posibilidad de extender la funcionalidad de estos controles básicos o crear nuestros propios controles personalizados.

4.1.3.3 Service[32]

Los servicios (o service) son tareas no visibles que se ejecutan en segundo plano, incluso cuando la actividad asociada no se encuentra en primer plano. Tiene un hilo propio de programación (aunque no se pueden ejecutar solo), lo que permite llevar a cabo cualquier tarea, por pesada que sea. No necesita interfaz gráfica, a no ser que se pida explícitamente.

Los servicios pueden realizar cualquier tipo de acciones, por ejemplo actualizar datos, lanzar notificaciones, o incluso mostrar elementos visuales (p.ej. activities) si se necesita en algún momento la interacción con del usuario.

Conviene aclarar que en situaciones donde el sistema necesite memoria conservar un servicio siempre será menos prioritario que la actividad visible en pantalla, aunque más prioritario que otras actividades en segundo plano. Dado que el número de actividades visibles es siempre reducido, un servicio solo será eliminado en situaciones de extrema necesidad de memoria.

4.1.3.4 Intent[33]

Un “intent” es el elemento básico de comunicación entre los distintos componentes Android. Se pueden entender como los mensajes o peticiones

[31] Salvador Gómez Oliver. **“***Curso de Programación Android”,* Madrid, 2011.
[32] Jesús Tomás Girónes. **“***El Gran Libro de Android”,* Primera edición, Marcobo.S.A, Barcelona-España, 2011.
[33] Salvador Gómez Oliver. *“Curso de Programación Android”,* Madrid, 2011.

que son enviados entre los distintos componentes de una aplicación o entre distintas aplicaciones. Mediante un “intent” se puede mostrar una actividad desde cualquier otra, iniciar un servicio, enviar un mensaje, iniciar otra aplicación, etc.

4.1.3.5 Broadcast Receiver[34]

Un “broadcast” receiver es un componente destinado a detectar y reaccionar ante determinados mensajes o eventos globales generados por el sistema (por ejemplo: “Batería baja”, “SMS recibido”, “Tarjeta SD insertada”, etc.) o por otras aplicaciones (cualquier aplicación puede generar mensajes “intents”, “broadcast”, etc).

4.1.3.6 Content Provider[35]

Un “content provider” es el mecanismo que se ha definido en Android para compartir datos entre aplicaciones. Mediante estos componentes es posible compartir determinados datos de nuestra aplicación sin mostrar detalles sobre su almacenamiento interno, su estructura, o su implementación. De la misma forma, nuestra aplicación podrá acceder a los datos de otra a través de los “content provider” que se hayan definido.

4.1.3.7 Widget[36]

Los “widgets” son elementos visuales, normalmente interactivos, que pueden mostrarse en la pantalla principal (home screen) del dispositivo Android y recibir actualizaciones periódicas. Permiten mostrar información de la aplicación al usuario directamente sobre la pantalla principal.

4.2 APLICACIÓN EN ANDROID

Debido a la necesidad que tienen las personas con discapacidad se seleccionó en diseñar una aplicación para teléfonos inteligentes con

[34] Salvador Gómez Oliver. *“Curso de Programación Android”,* Madrid, 2011.
[35] Salvador Gómez Oliver. **“***Curso de Programación Android”,* Madrid, 2011.
[36] Salvador Gómez Oliver. **“***Curso de Programación Android”,* Madrid, 2011.

plataforma Android y de esta manera lo que se pretende es mejorar su estilo de vida.

Los teléfonos móviles han ido evolucionando al punto que se puede hacer casi cualquier cosa con ellos. Esto es de gran importancia porque se puede ayudar mucho a personas con algún tipo de discapacidad a realizar actividades que normalmente lo realizan con mucho esfuerzo.

La aplicación desarrollada en esta tesis en un inicio estaba pensada para la persona que está a cargo de la persona con discapacidad para facilitar su labor .Pero, pensando en la necesidad de autonomía que tiene el discapacitado se añadió el control de la vivienda. Con esto el discapacitado puede: encender luces, abrir puertas, ventanas, persianas desde su teléfono Android.

Dicho esto y con los conceptos impartidos al inicio del capítulo a continuación se procederá a explicar el desarrollo de la aplicación para los fines antes mencionados.

4.2.1. Funcionamiento de la Aplicación

La aplicación desarrollada costa de las siguientes partes:

En primer lugar la aplicación puede controlar carga dentro de la vivienda, es decir, controla: encendido de luces, apertura de puertas, ventanas, persianas.

Además, la aplicación puede acceder a cámaras de vigilancia para el monitoreo de la vivienda y de la persona con discapacidad.

Y por último es capaz de mandar notificaciones ante situaciones de emergencia para lo cual consta de un sistema de envío de mensajes.

4.2.2. Estructura de la Aplicación

La aplicación consta de las siguientes actividades: la principal (main) llamada Domótica y tres actividades llamadas: Vivienda, VideoVigilancia, Seguridad. Además, unas actividades que sirven para configuración de Seguridad y una clase en la que se desarrolla una base de datos en SQlite.

Cada una de estas actividades tienen sus clases que se encuentran la carpeta “scr” .Cabe mencionar que para que una clase java se convierta en una actividad para Android se debe indicar en el archivo “AndroidManifest.xml”. Todas estas clases están contenidas en un paquete llamado “com.domotica” y el proyecto se llama Domotica.1 cómo se puede apreciar en la siguiente figura:

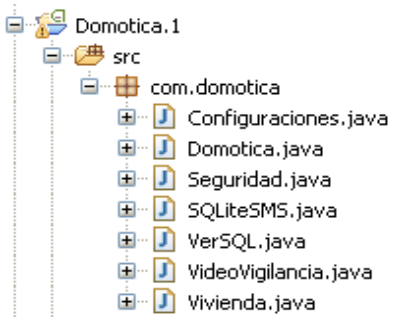


Figura 85. Estructura de las actividades

Por otra parte, tememos los recursos que van a ser usados por la aplicación. Estos se encuentran en la carpeta “res”. Los más importantes que encontramos son las imágenes almacenadas en las carpetas “drawable”, los ficheros “xml” que contienen las interfaces graficas contenidas en la carpeta “layout”.

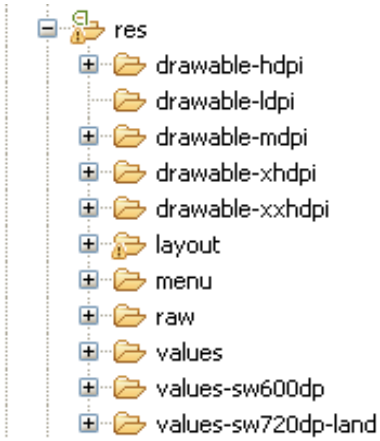


Figura 86. Directorio de recursos de la Aplicación.

4.2.3 Implementación de Actividades e Interfaces de Usuario

Las actividades en Android son usadas para gestionar los eventos producidos en la interfaz gráfica o usuario. En este apartado, detallaremos las actividades e interfaces que conforman nuestra aplicación así como su funcionamiento.

Como ya mencionamos cada una de las actividades que conforman la aplicación deben estar especificadas en el archivo “AndroidManifest.xml”, donde podemos ver, las actividades que componen la aplicación, así como, las características y opciones de las mismas.

En cuanto a las interfaces gráficas se irán explicando conjuntamente con su actividad. Estas interfaces son ficheros en “xml” que desempeñan un papel muy importante, ya que se trata de presentación de la Actividad al usuario. En la siguiente figura podemos ver un archivo “xml” de una de las interfaces que se implementan.

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/andr
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <WebView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/wb_control"
        android:focusable="true"/>
</LinearLayout>
```

Figura 87. Ejemplo de interface gráfica en "xml".

4.2.3.1 Domótica (Main Activity)

Esta actividad permite acceder a las demás actividades de la aplicación. Para qué sistema Android reconozca a esta actividad como su actividad principal debemos indicar en el archivo "AndroidManifest.xml", que tenga una categoría "Launcher", esto permitirá que la aplicación se inicie con esta actividad.

```xml
<activity
    android:name="com.domotica.Domotica"
    android:label="@string/app_name"
    android:theme="@android:style/Theme.Wallpaper">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Figura: Fragmento de AndroidManifest.xml que hace referencia a un MainActivity.

Como cada actividad debe tener un interfaz usuario, y en este caso la actividad posee un interfaz de tipo lista de objetos. Esta característica implica que Domótica herede de un "ListActivity" y no de un "Activity" como normalmente se crean las actividades.

Por lo tanto nuestra actividad Domótica tiene el siguiente aspecto:

Figura 88. Interfaz de Usuario de la Actividad Domótica.

Una característica adicional de la actividad es que el fondo es el "Wallpaper" del teléfono. Si se cambia el "Wallpaper" también se cambiara el fondo de la actividad.

4.2.3.2 Vivienda (Activity)

Esta actividad es la responsable de controlar carga dentro de una vivienda. Para ello, accedemos a una página de un servidor web (Placa Arduino). Consta básicamente de un pequeño explorador web en donde se carga la página del servidor.

La forma de referenciarlo en el archivo "AndroidManifest.xml", es con las siguientes líneas dentro de la etiqueta "aplication":

```
<activity
            android:name="com.domotica.Vivienda"
            android:label="@string/title_activity_vivienda" >
</activity>
```

Algo muy importante que resaltar es que para que la aplicación se pueda conectar a una red, y así poder acceder al servidor. Necesitamos un permiso del sistema y este permiso es el de acceso a internet. Para solicitar este permiso escribimos las siguientes líneas en el archivo "AndroidManifest.xml" dentro de la etiqueta "manifest"

```
<uses-permission android:name="android.permission.INTERNET" />
```

Su interfaz gráfica es un View del tipo “WebView” que lo podemos encontrar que la podemos encontrar en la carpeta “layout” con el nombre de controldomotico.xml

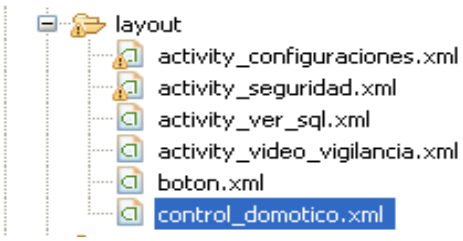


Figura 89. Archivo “xml” de la interfaz grafica de la Actividad Vivienda.

A continuación se muestra una imagen del funcionamiento de esta Actividad.

Figura 90. Interfaz de Usuario de la Actividad Vivienda.

4.2.3.3 VideoVigilancia (Activity)

Esta actividad permite acceder a cámaras de seguridad del tipo ip. Para de esta manera monitorear a la persona con discapacidad o vigilar zonas de la casa en las que se requiera este servicio. Al igual que la actividad anterior consta de pequeño explorador web en donde se cargan las imágenes provenientes de la cámara ip. Además, nos permite acceder a un máximo de 3 cámaras.

La forma de referenciarlo en el archivo “AndroidManifest.xml”, es similar a como se hizo para la actividad anterior. Para este caso sería:

```
<activity
android:name="com.domotica.VideoVigilancia"
android:label="@string/title_activity_video_vigilancia">
</activity>
```

En cuanto a la interfaz usuario consta de un “View” “Spinner” (similar a listas desplegables en otros lenguajes de programación) que nos permite seleccionar la cámara, un botón que nos permite comenzar la video vigilancia y un “WebView” que es en donde se carga las imágenes provenientes de la cámara ip. Esta la podemos encontrar en la carpeta “layout” con el nombre de activity_video_vigilancia.xml como se puede ver a continuación:

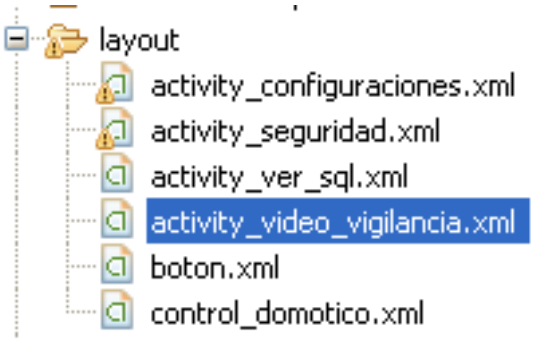


Figura 91. Archivo “xml” de la interfaz gráfica de la Actividad VideoVigilancia.

En la siguiente figura se muestra la interfaz gráfica de la actividad:

Figura 92. Interfaz de Usuario de la Actividad VideoVigilancia.

4.2.3.4 Seguridad (Activity)

Esta actividad permite enviar mensajes en situaciones de emergencia a distintas personas con solo presionar un botón. Para esto usa una base de datos en SQlite en donde se almacena los mensajes y los destinatarios. Tiene una capacidad máxima de envió de 5 mensajes.

Además, esta actividad cuenta con un menú en donde tenemos las opciones de configuración y ayuda. Mediante la opción configuración podemos modificar los valores en la base de datos.

Algo que destacar es que cuando ponemos en el cuerpo del mensaje las letras NE (No Enviar), el mensaje no se enviará al número correspondiente. Esto es una forma de limitar el envío de mensajes.

Para referenciar están actividad en el archivo “AndroidManifest.xml”, lo hacemos igual que las anteriores actividades.

Por otra parte, para que la aplicación tenga la cualidad de envió de mensajes debe utilizar un permiso del sistema como en el caso de las actividades que necesitaban conectarse a una red. Este permiso es el de envió de mensajes que se solicita de con las siguientes líneas en el archivo

"AndroidManifest.xml", dentro de la etiqueta "manifest" como era el caso del permiso de acceso a internet.

```
<uses-permission android:name="android.permission.SEND_SMS" />
```

En cuanto a la interfaz usuario consta de un botón de emergencia que al presionarlo hará posible en envió de los mensajes a los destinatarios. Al igual que las anteriores interfaces se encuentra ubicado en carpeta "layout" con el nombre de activity_seguridad.xml.

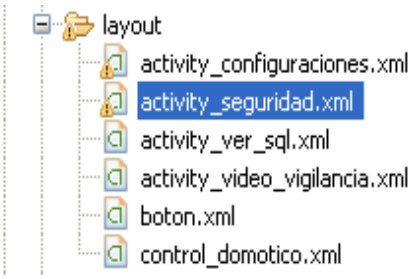


Figura 93. Archivo xml de la interfaz gráfica de la Actividad Seguridad.

En la siguiente figura se muestra la interfaz gráfica de la actividad.

Figura 94. Interfaz de Usuario de la Actividad Seguridad.

Al presionar en "SI" se procede al envió de los mensajes.

Para que esta actividad funcione adecuadamente necesita de la actividad configuraciones y de la base de datos antes mencionadas que precederemos a detallarlas a continuación.

4.2.3.5 SQLiteSMS (Clase)

Android tiene incorporado un gestor de base de datos SQLite. SQLite es un motor de base de datos que se ha ido popularizando en los últimos años dado que maneja archivos de poco tamaño, no necesita ejecutarse en un servidor y, además, es de código libre.[37]

Esta base de datos consta de una tabla de 3 columnas en donde la primera columna corresponde a un identificador (ID), la segunda corresponde al cuerpo del mensaje a ser enviado y la tercera se guarda el número telefónico del destinatario del mensaje.

ID	***Mensaje***	***Destinatario***
1	Emergencia 1	Número de celular 1
2	Emergencia 2	Número de celular 2
3	Emergencia 3	Número de celular 3

Tabla 6. Ejemplo de la base de datos.

Para poder acceder a esta base de datos necesitamos la actividad Configuraciones.

4.2.3.6 Configuraciones (Activity)

La actividad configuraciones nos permite ingresar, modificar, buscar y eliminar información dentro de nuestra base de datos. Podemos acceder a esta actividad desde la actividad Seguridad al presionar en el icono de configuraciones después de pulsar la tecla menú en nuestro dispositivo Android como se indica en la siguiente figura:

[37] Manuel Báez, Álvaro Borrego, Jorge Cordero, Luis Cruz, Miguel González, Francisco Hernández, David Palomero, José Rodríguez de Llera, Daniel Sanz, Mariam Saucedo, Pilar Torralbo, Álvaro Zapata. **"***Introducción a Android",* Madrid, 2011.

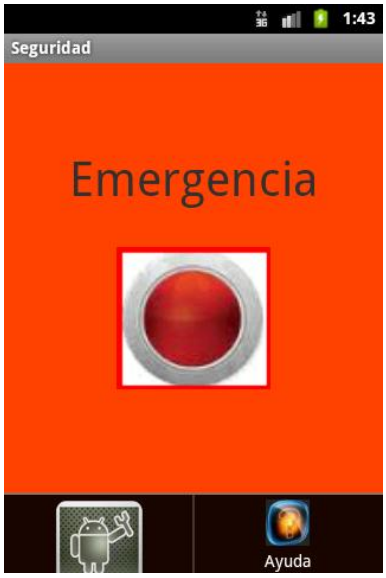


Figura 95. Acceso a la actividad Configuraciones.

La Actividad Configuraciones tiene una interfaz de usuario en donde podemos introducir tanto el mensaje como el número telefónico del destinatario y su ID si queremos buscar, editar o borrar esa información de la base de datos. Consta básicamente de los siguientes: “Views”, “TextEditors” y botones. Está ubicado en la carpeta “layout” con el nombre de activity_configuraciones.xml. En la siguiente figura se puede ver la Actividad en funcionamiento.

Figura 96. Interfaz de Usuario de la Actividad Configuraciones.

El procedimiento para referenciar está actividad en el archivo “AndroidManifest.xml”, es el mismo que en los casos anteriores.

4.2.3.7 VerSQL (Activity)

Esta actividad nos permite ver la información que está dentro de nuestra base de datos. Para acceder a esta actividad pulsamos el botón ver de la actividad Configuraciones. Una característica diferente a las anteriores actividades es el aspecto de su interfaz gráfica. La cual tiene la apariencia de un dialogo. Consta básicamente de un "texView" donde se va a mostrar la información. El nombre del archivo de su interfaz gráfica es activity_ver_sql.xml.

Figura 97. Interfaz de Usuario de la Actividad VerSQL.

Para obtener esta interfaz en el momento de referenciar la actividad en el archivo "AndroidManifest.xml" lo hacemos con las siguientes líneas.

```
<activity
            android:name="com.domotica.VerSQL"
            android:label="@string/title_activity_ver_sql"
            android:theme="@android:style/Theme.Dialog" >
</activity>
```

Como podemos ver solo agregamos una línea adicional en la cual le decimos que la actividad tenga la forma de un dialogo.

4.2.3.8 AndroidManifest.xml

Este archivo básicamente es la medula espinal de toda aplicación, ya que en él se declararán tanto los componentes de Android utilizados, así como los recursos del sistema que la aplicación necesita.

En nuestro caso, el manifiesto contendrá la declaración de las actividades de las que se compone la aplicación, indicando el modo y las características de la ejecución de cada una de ellas. Además, los permisos de los que dispone la aplicación, en este caso, los permisos necesarios para la correcta ejecución de la aplicación son la conexión internet y permiso de envió de mensajes como se pude apreciar a continuación.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.domotica"
    android:installLocation="preferExternal"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="10" />

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.SEND_SMS" />

    <application
        android:allowBackup="true"
        android:debuggable="true"
        android:icon="@drawable/icono_domotica"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.domotica.Domotica"
            android:label="@string/app_name"
            android:theme="@android:style/Theme.Wallpaper">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
```

Figura 98. Fragmento del archivo “AndroidManifest.xml” de nuestra aplicación.

CAPITULO 5

IMPLEMENTACION DEL PROTOTIPO

5.1 INTRODUCCION

Una vez concluido con el diseño de los sistemas de control, se tuvo la necesidad de comprobar su funcionamiento. Para lo cual se procedió a construir un prototipo en donde se pueda apreciar su funcionalidad.

En este prototipo se diseña todo lo necesario para que los sistemas desarrollados funcionen correctamente. Aquí se seleccionan todos los dispositivos que van a ser utilizados, se construyen placas y una maqueta para montar todos los elementos.

A continuación se explica cada uno de los elementos que conforman este prototipo.

5.2 PLACA ARDUINO UNO

En el capítulo donde se desarrolla la aplicación de visión artificial se explicó que este sistema se comunica externamente con comunicación serial y los datos eran recogidos por una placa Arduino uno.

Ha llegado el momento de hablar un poco más sobre Arduino y comenzaremos diciendo que es una plataforma para prototipos electrónicos de código abierto. Su hardware es libre por lo tanto puede reutilizarse e incluso mejorarse. Fue pensado para todos los entusiasta de la electrónica.[38] Arduino consta de un entorno de desarrollo integrado IDE muy cómo e flexible que tiene una combinación curiosa del lenguaje java con C++.

A continuación se muestra una vista de este entorno.

[38] *Curso de supervivencia con Arduino* by Juan Gregorio Regalado Pacheco, 44 p.

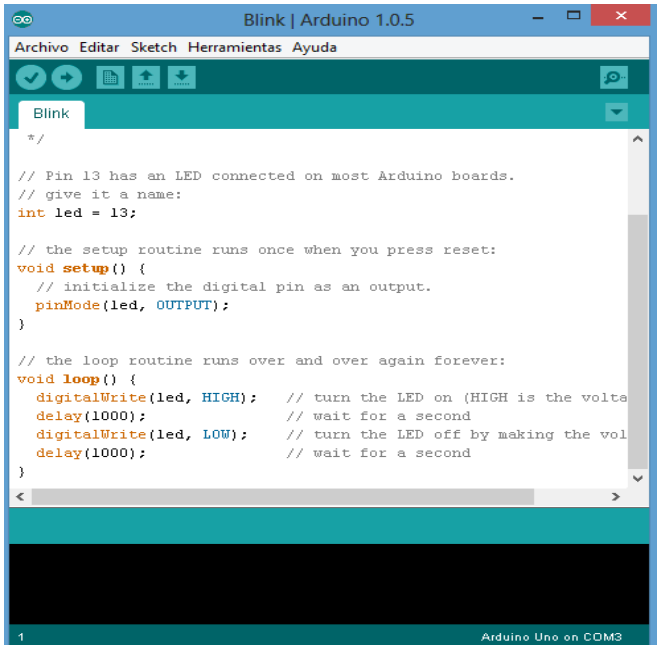
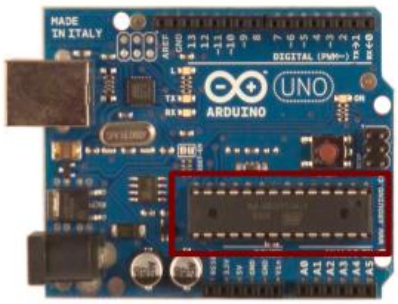


Figura 99. IDE de Arduino.

Hay muchas placas Arduino en el mercado desde las más pequeñas hasta las más grandes, pero la que empleamos en esta parte de la tesis es la Placa Arduino Uno que tiene las siguientes características.



Microcontrolador	Atmega328
Voltaje de operación	5V
Voltaje de entrada (Recomendado)	7 – 12V
Voltaje de entrada (Límite)	6 – 20V
Pines para entrada- salida digital.	14 (6 pueden usarse como salida de PWM)
Pines de entrada analógica.	6
Corriente continua por pin IO	40 mA
Corriente continua en el pin 3.3V	50 mA
Memoria Flash	32 KB (0,5 KB ocupados por el bootloader)
SRAM	2 KB
EEPROM	1 KB
Frecuencia de reloj	16 MHz

Figura 100. Características de Arduino Uno.

La función de esta placa es recoger los datos provenientes del sistema de visión artificial a través de comunicación serial. Una vez que recibe un dato verifica a que código corresponde y al existir coincidencia manda a ejecutar una acción. Estas acciones básicamente son señales para activar relés para cargas on/off, control PWM en servomotores para simular apertura/cierre de puertas y ventanas.

La razón para el elegir esta placa es su facilidad en su programación. Dado que su IDE tienen varias librerías que permiten hacer cosas muy complicadas en un menor tiempo. Y también, por experimentar nuevas opciones para construir prototipos electrónicos.

5.3 MICROCONTROLADOR 16F877A EMPLEADO PARA COMANDOS DE VOZ

La forma con la cual se comunica el sistema de comandos de voz es comunicación serial. Todos datos enviados son recogidos por el micro controlador 16f877A de microchip. Este micro está programado en lenguaje basic cuyo entrono de desarrollo es Proton IDE.

Se función recibir datos, compararlos y enviar señales para activar relés y controlar servomotores con el mismo objetivo que en el punto anterior.

5.3.1 Asignación y distribución de los pines del Pic 16F877A

Se ha asignado distribuido los pines de tal forma que nos permitan enviar señales de entrada y salida según sea la necesidad de la aplicación, por ello a continuación se tiene la asignación de pines.

PIC 16F877A			
Pin	Nombre	Configuración	Descripción
1	MCLR	Entrada	Puede reiniciar el microcontrolador
2	RA0	Entrada	Reserva para sensor 1

3	RA1	Entrada	Reserva para sensor 2
4	RA2	Entrada	Reserva para sensor 3
5	RA3	Entrada	Reserva para sensor 4
6	RA4	-	
7	RA5	-	
8	RE0	-	
9	RE1	-	
10	RE2	-	
11	VCC	-	Alimentación VCC +
12	GND	-	Alimentación VCC -
13	OSC1	-	Cristal de 4MHZ
14	OSC2	-	
15	RC0	-	
16	RC1	-	
17	RC2	-	
18	RC3	-	
19	RD0	-	
20	RD1	-	
21	RD2	-	
22	RD3	-	
23	RC4	-	
24	RC5	-	
25	RC6	Salida	Puerto serial
26	RC7	Entrada	
27	RD4	-	
28	RD5	-	
29	RD6	-	

30	RD7	-	
31	GND	-	
32	VCC	-	
33	RB0	Salida	Señal de control para luminaria
34	RB1	Salida	Señal de control para servomotor 1
35	RB2	Salida	Señal de control para servomotor 2
36	RB3	Salida	Reserva
37	RB4	Salida	Reserva
38	RB5	Salida	Señal de control para ventilador
39	RB6	Salida	Señal de control para alarma
40	RB7	-	

Tabla 7. Asignación de pines para el microcontrolador 16F877A

5.4 RED DE COMUNICACIÓN CON LA APLICACIÓN ANDROID

Tal vez este sistema de comunicación de datos sea un poco más complicado que en los anteriores sistemas dado que la forma en que se comunica la aplicación en “Android” es mediante “Wifi” que es un mecanismo de conexión inalámbrica y la forma de transmisión de datos es mediante el protocolo TCP-IP. Entonces, para poder recibir batos o enviar datos a la aplicación se optó en construir una red doméstica LAN. Para lo cual se utilizó un “Router” con un “Switch” integrado de la marca “TPlinK”.

El router va trabajar como un punto de acceso a los demás elementos de la red. Los elementos que conforman la red son: el celular con sistema operativo “Android”, la cámara ip de la marca DLink, una placa Arduino nano con su “Shield Ethernet” y el “router TPLink”. Cómo se muestra en la siguiente figura:

Figura 101. Estructura de la red LAN.

La red mostrada en la figura anterior es muy sencilla dado que nuestro objetivo es mostrar funcionamiento de la aplicación en “Android”. Pero se la puede ampliar cuanto se quiera agregando más cámaras ip, más placas Arduino, más routers, etc. En fin se puede agregar cualquier elemento que se desee controlar o acceder. Lo único que hay que hacer es ampliar la funcionalidad de la aplicación en “Android” para todos los elementos de la red.

Algo que resaltar es que la placa Arduino nano es una placa con iguales características que la placa Arduino Uno pero mucho más pequeña y su costo es menor. Además, posee modulo denominado “Ethernet Shield” que nos permite conectarse a una red razones por la cual esta placa se usa en este sistema.

La finalidad de la placa Arduino nano es la de un servidor Web que contiene página web que permite que el celular pueda ejercer el control de las cargas deseadas. Una vez seleccionado el control a ejercer envía las señales correspondientes que como en el caso de las placa Arduino uno es la de activar relés y controlar servomotores.

5.5 PLACAS DE POTENCIA

En estas placas se encuentran todos los circuitos necesarios para gestionar las cargas se van a controlar en este prototipo. Las cargas que

controlaremos son: encendido/apagado de luminarias, encendido/apagado de un ventilador, apertura/cierre de puertas y ventanas.

Tanto para las luminarias como para el ventilador usamos relés y su circuito de conmutación es el siguiente:

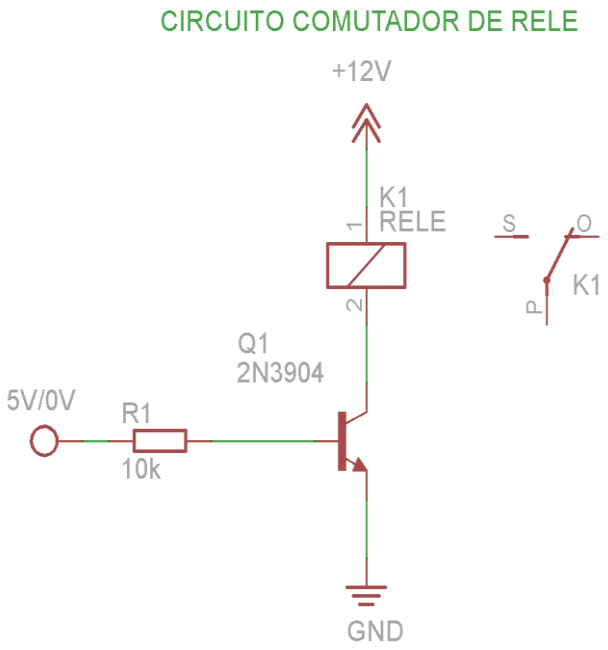


Figura 102. Circuito conmutador de relé.

Para el caso de puertas y ventanas usamos Servomotores con esto simulamos su apertura y cierre. Para este caso el control se hace mediante PWM.

Figura 103. Servomotor.

Conocido como se hace el control se procedió a construir 2 placas que nos van a servir para nuestro objetivo. El software que se emplea para este fin es Eagle 6.4.

5.5.1 Placa para Gestión de Cargas desde Visión Artificial y Android

Como describimos anteriormente tanto la aplicación de visión artificial y Android se comunican a placas Arduino y estas envían las mismas señales de control. Por lo tanto se construyó una placa que reciba estas señales y ejecute gestión en las cargas.

A continuación se muestras el esquemático, su vista en PCB y su acabado final.

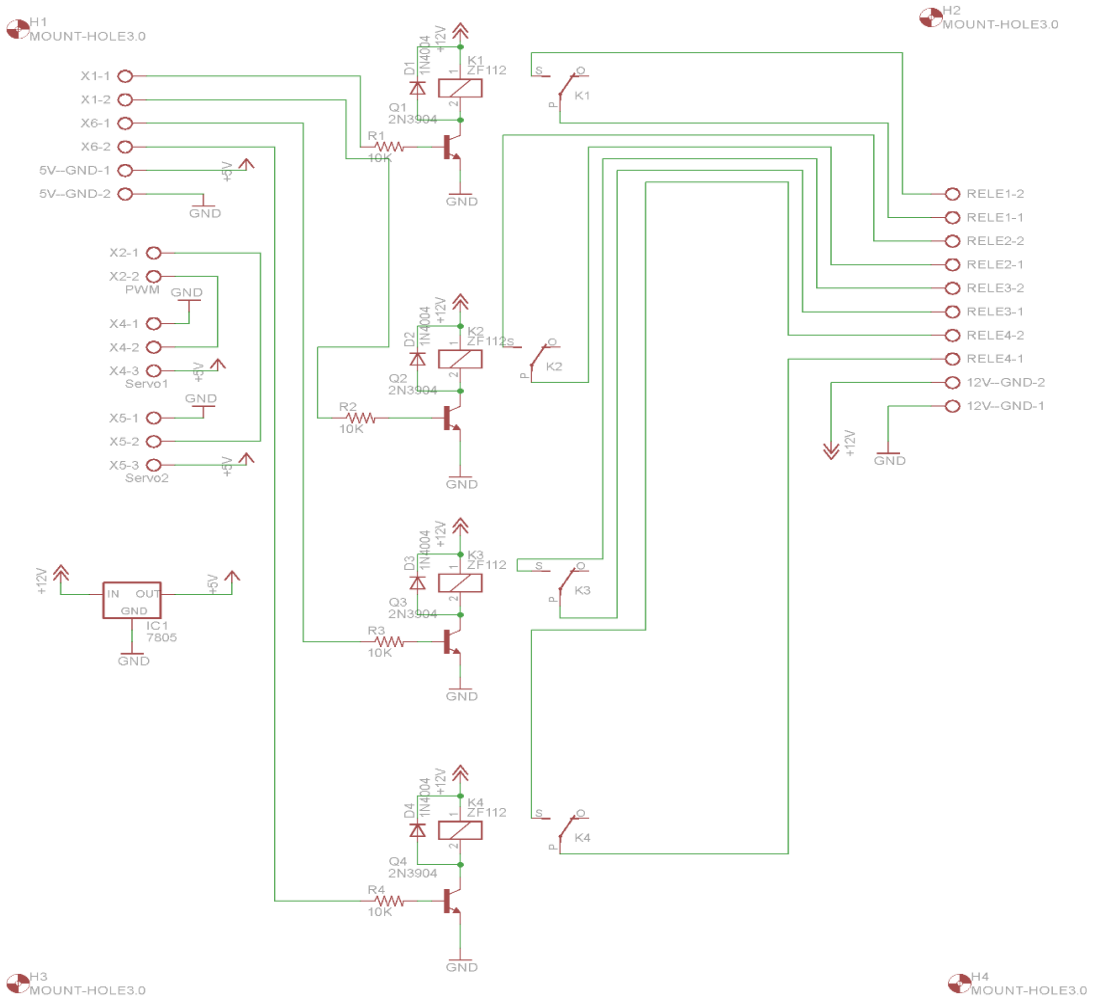


Figura 104. Esquemático de la placa 1.

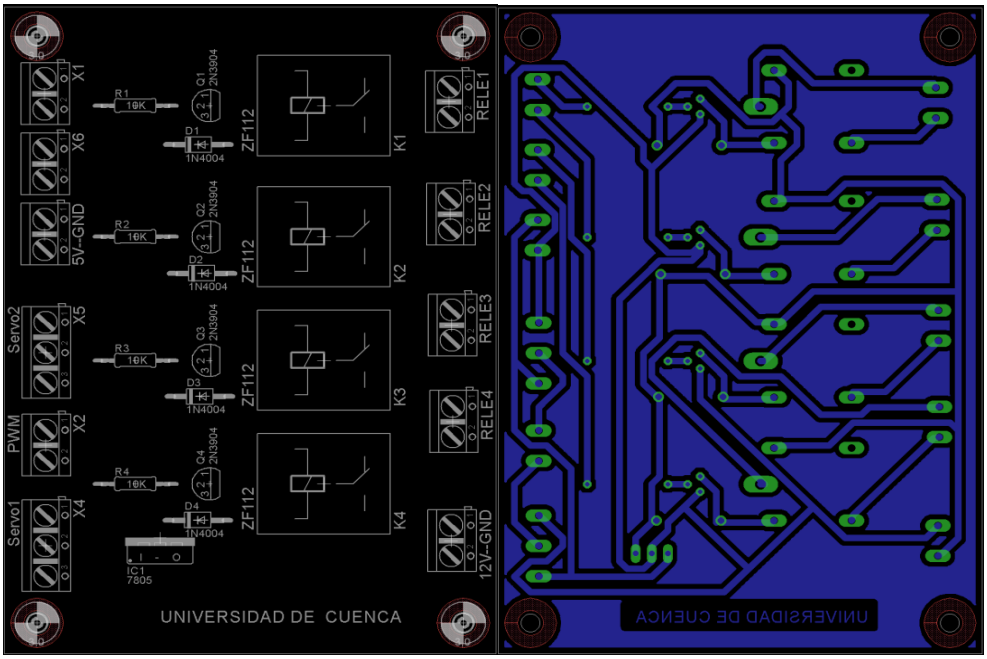


Figura 105. Vista del PCB de la placa 1.

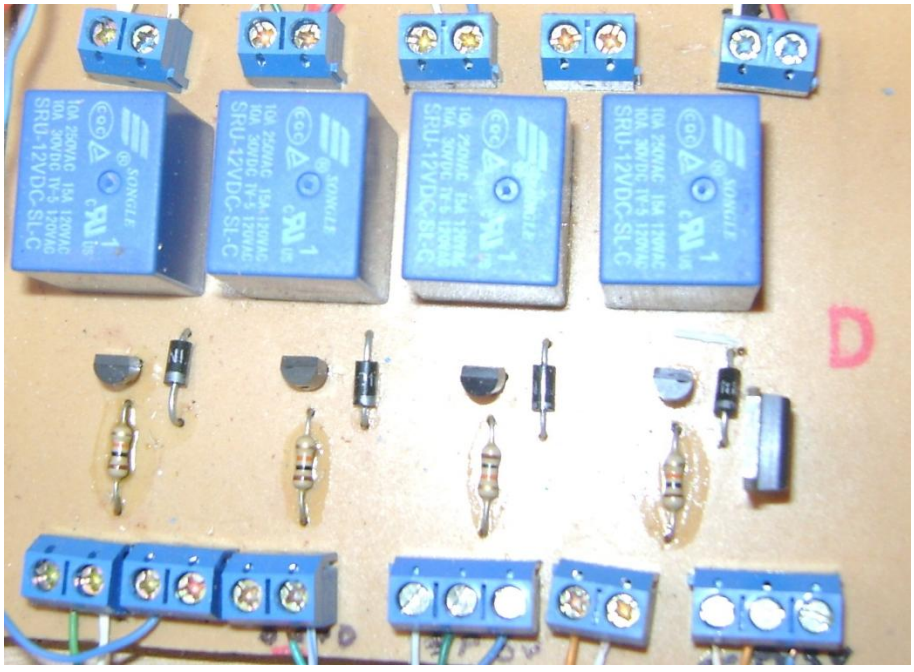


Figura 106. Acabado final de la Palca 1.

Para el funcionamiento de la maqueta usaremos dos relés y las salidas para los servos quedando los relés restantes de reserva para funciones aplicaciones.

5.5.2 Placa para Gestión de Cargas desde Comandos de Voz

Dado que la aplicación de comandos de voz se comunica con un microcontrolador 16F877A se procedió a construir una placa que incorpore al microcontrolador, la comunicación serial y la parte de potencia para la gestión de la cargas.

La placa podrá tener el control de las siguientes cargas: una luminaria, dos servomotores, un ventilador y una sirena o buffer. Además, tiene reservas para cuatro entradas y dos salidas que sirven para cuando se necesite hacer una ampliación de cargas a controlar o sensores que ayuden a mejorar el sistema de comandos de voz.

A continuación se indica el esquemático de esta placa con su respetivo PCB y su acabado final.

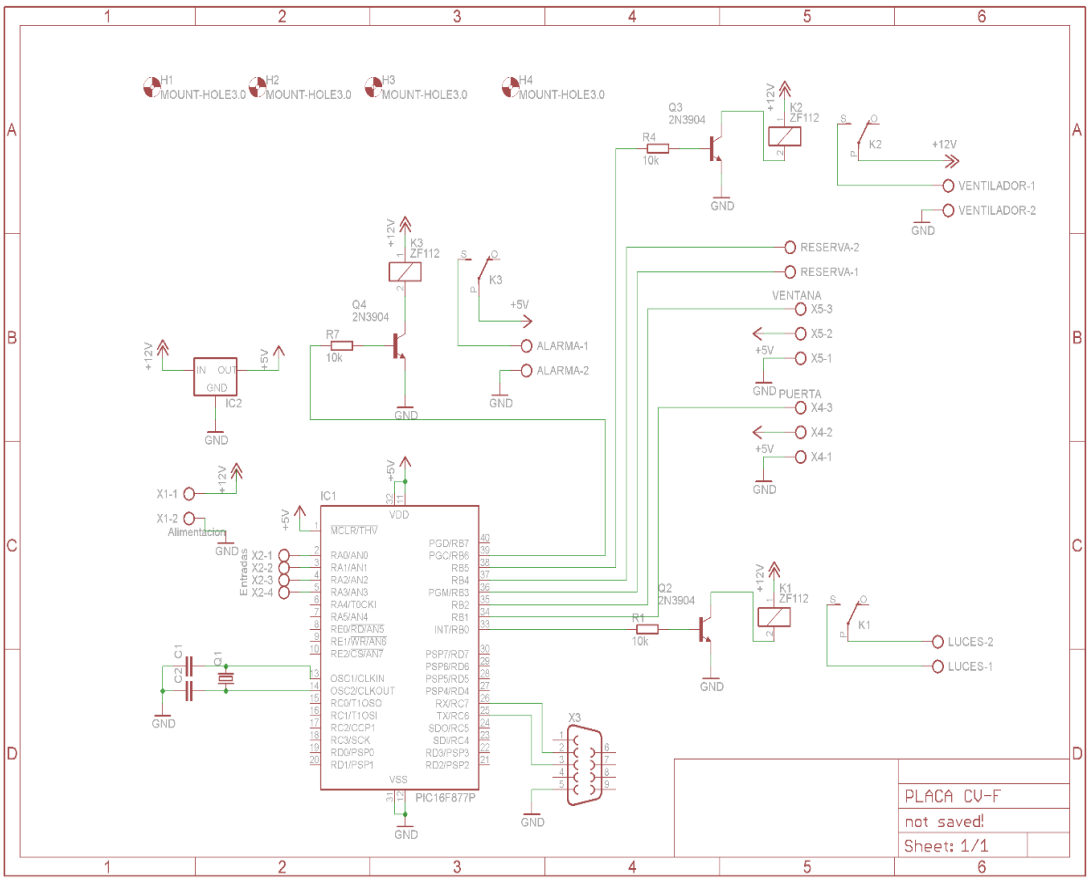


Figura 107. Esquemático de placa 2.

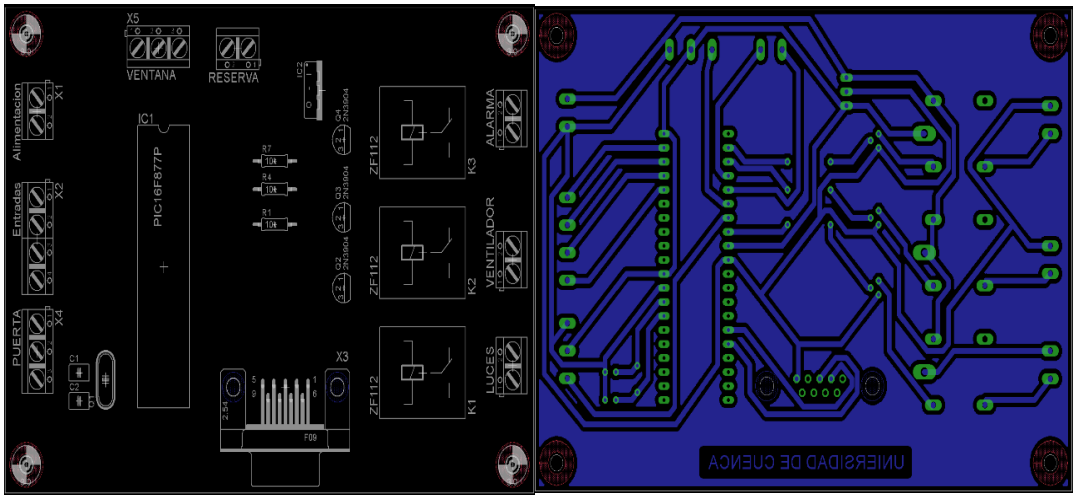


Figura 108. Vista del PBC de la Placa2.

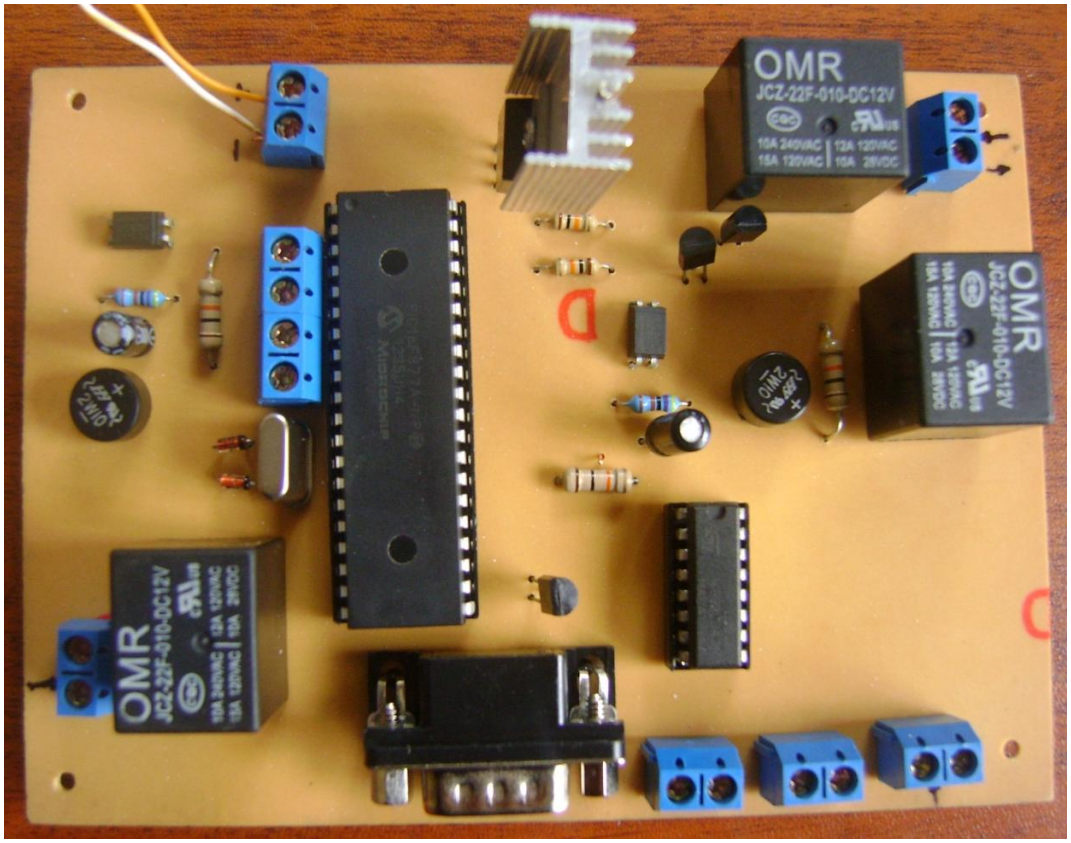


Figura 109. Acabado final de la Palca 2.

5.6 CONSTRUCCION DE LA MAQUETA

Se plantea la elaboración de un prototipo para control domótico de una vivienda de un solo ambiente, por lo cual se hace necesaria la construcción

de una maqueta que represente una vivienda, en la cual se colocara los componentes y conjuntamente con las aplicaciones de control se simulará a una vivienda domótica apta para personas con discapacidad.

Figura 110. Maqueta, para simular vivienda domotica.

5.6.1 Colocación de Elementos en la Maqueta

A continuación se muestra a través de fotografías la colocación de los elementos dentro de la maqueta. La cual incluye la colocación de: luminarias, ventilador, servos y placas de control con sus respectivas pruebas.

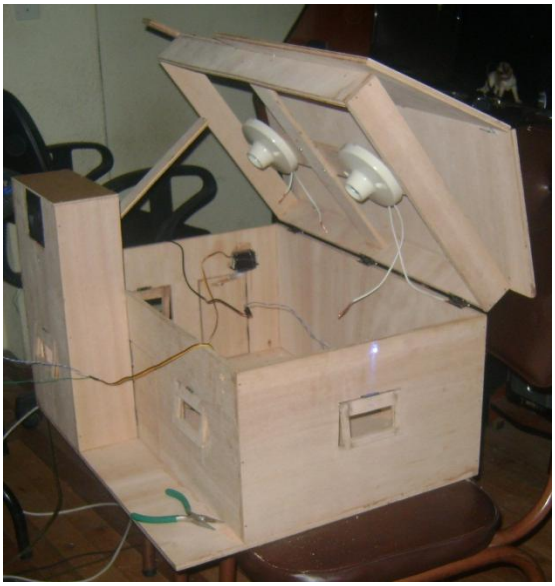


Figura 111. Montaje de servos, luminarias y ventilador.

Figura 112. Prueba de placas de potencia para su posterior montaje.

Figura 113. Control de cargas mediante comandos de voz y aplicación en Android.

Figura 114. Control de cargas mediante visión artificial.

Figura 115. Prototipo concluido para conexión a los tres sistemas de control desarrollados.

5.7 DESCRIPCIÓN DEL PROTOTIPO

Como se puede observar en las imágenes del punto anterior ha sido necesario ir probando y verificando el funcionamiento de todos los elementos que conforman el prototipo.

Como resultado final se tiene el prototipo que está listo para ser conectado o enlazado a los siguientes sistemas de control:

1. Sistema de control mediante visión artificial a través de señales.
2. Sistema de control mediante comandos de voz.
3. Sistema de control y vigilancia por medio de aplicación en Android.

A continuación se desarrolla un cuadro que detalla que cable y a donde de ir conectado.

CABLE		DESCRIPCION
TIPO	COLOR	
Flexible #18	Rojo	Alimentación 12V +
Flexible #18	Negro	Alimentación 12V -
Multipar	Café	Alimentación 5V +
Multipar	Café-blanco	Alimentación 5v -

Cable serial	Gris	Al puerto serial PC
Multipar	Tomate	Señal PWM para el servo de la puerta
Multipar	Tomate-blanco	Señal PWM para el servo de la ventana
Multipar	Verde	Señal de control para la luminaria
Multipar	Verde-blanco	Señal de control para el ventilador
Multipar	Celeste-blanco	Referencia GND

Tabla 8. Designación de cables para la conexión

CAPITULO 6

CONCLUSIONES Y RECOMENDACIONES

6.1 CONCLUSIONES

- El presente proyecto nos sirvió para aprender que todo en la vida es posible y cuando se quiere se puede. Sobre todo cuando la finalidad del proyecto es ayudar a personas con discapacidad a mejorar su estilo de vida. Y aunque a los sistemas desarrollados se los puede optimizar, las ideas mostradas en esta tesis pueden ser motivo de seguir investigando, trabajando, aprendiendo no solo a los que desarrollamos esta tesis sino a otras personas interesados en el área de la domótica.

- Con la aplicación control mediante comandos de voz se ha logrado que personas que tengas dificultad a la hora de movilizarse, puedan manipular algunas cargas desde un lugar fijo.

- Los comandos de voz permite que personas con discapacidad visual puedan hacer uso de su voz, a la hora de intentar manipular una carga eléctrica, con lo cual logramos que tengas cierta autonomía al momento de la activación o desactivación de un dispositivo eléctrico.

- Con la utilización de la aplicación de comandos de voz, se logra que la persona discapacitada o persona sin ningún grado de discapacidad puedan manipular de una manera sencilla y guiada, varias cargas que son comunes en una vivienda familiar, ya que el diseño estada basado en un criterio universal.

- El software para reconocimiento de voz “Dragon” permite que la aplicación por comandos de voz pueda ser usada por varias personas, ya que permite crear perfiles distintos para cada usuario y

con un breve entrenamiento se logra tener el control adecuado de la aplicación.

- Gracias al desarrollo del proyecto se adquirió muchos conocimientos que es un objetivo intrínseco de toda tesis y que a lo largo de la vida pueden ser muy útiles para un sin número de proyectos. Así como adentrarnos a los campos de la domótica, visión artificial, aplicaciones móviles bajo plataforma “Android”.

- Las aplicaciones mostradas en esta tesis pueden ser implementadas de forma independiente. Puesto su funcionamiento no está entrelazado permitiendo así autonomía. Y su destino puede ser no solo para personas con discapacidad sino para cualesquier persona.

- El uso de software “Labview” junto con su “toolkit” de visión artificial es una herramienta muy eficiente que nos permite hacer aplicaciones de visión en muy poco tiempo obteniendo resultados muy eficientes. Además que su entrono de programación es fácil de entender y no es necesario de conocimientos avanzados en programación.

- Para que los sistemas de visión artificial funcionen correctamente existen ciertos factores de vital importancia como son: uniformidad en la iluminación, contraste, adquisición de una imagen de calidad. Todos estos factores deben ser considerados cuando se diseñen y monten proyectos de tengan como centro el procesamiento de imágenes.

- El uso de operaciones lógicas de las imágenes junto con la aplicación de filtros gaussianos, transformación a escala de grises y las herramientas de búsqueda de patrones, permitió cumplir con tarea fundamental del software de visión artificial.

- En cuanto a proyectos que tengan como entradas señales ya sea digitales o analógicas se debe realizar el adecuado procesamiento de la señal para de esta forma eliminar en gran medida el ruido que hace que los sistemas tengan errores.

- El sistema de visión para información de obstáculos y conjuntamente con la señalización de obstáculos con franjas de colores, permite que la persona con discapacidad visual, escuche y en base a eso tome una decisión de cambio de trayectoria.

- El sistema de información de obstáculos usando colores funciona con eficiencia ya que se compara patrones de colores y no es necesario identificar forma o tamaño. Por ello el sistema facilita el desplazamiento del no vidente dentro de la vivienda y da mayor autonomía a dicha persona.

- La realización de la aplicación en “Android” nos enseñó a manejar ciertas tecnologías que se desconocía por completo así como el uso de herramientas gratuitas con un gran potencial, como es el caso de Eclipse. Programar una aplicación en android requiere de mucho esfuerzo y paciencia sobre todo si no se tiene los conocimientos de programación que se necesita.

- Android nos brinda un gran abanico de posibilidades y un gran campo para explotarlo cuyo límite es nuestra imaginación. Podemos hacer todo tipo de control con esta plataforma móvil lo único que hay que hacer es intentarlo.

- El uso de microcontroladores y placas “Arduino” que son fáciles de programar y bajo costo, evitan los gastos elevados que representaría utilizar PLCs.

- Las aplicaciones desarrolladas permiten que personas que posean cierto grado de discapacidad, puedan tener un grado de autonomía y así no depender de una segunda personal al momento de querer realizar alguna actividad dentro del hogar.

- Con la investigación y desarrollo de alternativas para un control de una vivienda domótica orientada a discapacitados, se obtiene como resultado un incremento en la comodidad y facilidad de manejo de cargas básicas de un hogar, con ello las personas con diferentes tipos de discapacidad están formando y sintiéndose parte de los avances tecnológicos.

6.2 RECOMENDACIONES

- Se recomienda mejorar el sistema de iluminación del sistema de visión artificial. Para que se adapte con la luz ambiente haciendo uso de un control PID con la finalidad de mejorar la calidad de imagen adquirida y no la influya cambios de iluminación.

- Para tener mayor efectividad en el control domótico mediante comandos de voz, se debería generar un ambiente aislado de ruido y otras voces ajenas al individuo que desea realizar la orden a través de voz.

- Para lograr mayor autonomía y distancia de control se recomienda usa micrófonos inalámbricos ya sean estoy con tecnología bluetooth o wifi, con ello el discapacitado por optar controlar desde distintos puntos.

- Se recomienda utilizara un micrófono profesión para obtener mejores resultados a la hora de manipular cargas mediante voz y la colocación correcta del micrófono.

- Se debe analizar a fondo todas las herramientas de visión artificial que posee “Labview” para desarrollar aplicaciones en menor tiempo.

- El tiempo de análisis de las imágenes depende del procesador de la computadora donde se esté ejecutando el programa, es aconsejable ejecutarlo en computadoras más actuales.

- Si se requiere de una mejor resolución de las imágenes capturadas para aplicaciones en donde se requiera una mejor calidad. Se recomienda utilizar cámaras industriales que presentan mayor velocidad de cuadros tomados por segundo y menor ruido.

- El sistema de información de obstáculos podría ser mejorado para que no solamente de información sino que pueda guiar a la persona con discapacidad visual.

- Para una aplicación real se debe realizar mayor trabajo en el procesamiento de los colores, para obtener mayor eficiencia del sistema.

- Se remienda mejorar la página web construida en Arduino nano para la gestión de cargas desde “Android”. Para lo cual se puede buscar otro elemento que permita hacer páginas más sofisticadas o en su defecto un elemento con más memoria RAM y mejor acceso de red que fue un limitante al usar la placa Arduino nano.

- Antes de empezar a realizar una aplicación en “Android”, se recomienda aprender programación orientada a objetos, el lenguaje Java y sobretodo entender muy bien el ciclo de vida de una actividad.

- Se recomienda mejorar la red de la aplicación Android para que se pueda gestionar las cargas no solo desde del interior de la red LAN sino a través de internet.

- Las aplicaciones desarrolladas están diseñas para permitir la ampliación de las mismas, por ello se recomienda continuar la investigación en los tres campos que abarca este proyecto, para con ello contribuir de mejor manera al desarrollo de sistemas que ayuden a personas discapacitadas, y así hacerles parte del desarrollo de nuevas tecnologías para facilitar el vivir diario.

BIBLIOGRAFÍA

[1] Acosta Luque Pablo Moisés, Rivadeneira Astudillo Juan José. "*Diseño e Implemetacion de un Sistema IVR para Telecontrol Domotico por medio de un Telefono Movil*". Salgolquí- Ecuador, 2006.

[2] Acurio Mendez Eliana Maribel, Encarnacion Aguila Diana Maribel. "*Diseño y Construcción de un Módulo Didáctico de Visión Artificial Orientado al Control de Calidad de Llenado de Botellas de Vidrio con Diferente Tipo de Líquido*". Quito, 2011.

[3] Arévalo Molina Pedro José, Herrera Anda Santiago Andrés, "*Diseño, Construcción y de un Control Robot Industrial con Arquitectura Antropomórfica dotado con Visión Artificial*". Quito-Ecuador, 2010.

[4] Basantes Ortiz Juan Carlos, Torres Tufuño Freddy Ricardo, "Desarrollo de un Sistema de Control para un Brazo Robótico Mediante Adquisición y Procesamiento de Imágenes"

[5] Campos,L - *Tecnología Alternativa en el desarrollo del aprendizaje y la comunicación* -Camac - Argentina - CIIEE 98

[6] Cano Calabuig, Raúl. "*Diseño e implementación de una aplicación domótica en Android reconfigurable a partir de un XML*". Valencia, 2010

[7] Consejería de la economía e innovación tecnológica. "*La Domotica como Solucion cel Futuro*". Madrid ,2007.

[8] Consejo Nacional de Discapacidades, (2012) CONADIS, Disponible en http://www.conadis.gob.ec/provincias.php

[9] Curso de domótica e inmótica, JUAN ANTONIO, 2009.

[10] Deloitte and Touche (2003): *Access to Assistive Technology in the European Union,* European Commission, Directorate-General for Employment and Social Affairs.

[11] E. Oviedo, S. Velásquez, C. Isaza. *Sistema de control de luz a partir de comandos de voz usando RNA*. II Congreso De Microelectrónica Aplicada. Argentina. 2011.

[12] Fernandez, L. (s.f.). *Aportaciones a la mejora de los sistemas de reconocimiento.* Recuperado el 5 de Febrero de 2013, de http://www/gts.tsc.uvigo.es/-ldocio/PhdThesis.pdf

[13] Fonseca, F. F. (2010). *"Diseno y construcción de un sistema para el control de encendido y apagado de dispositivos eléctricos por medio de comandos de voz"*. Quito.

[14] Gaitan, E. G. (2009). *Paseo por Visual Basic 2008*. Granada, Nicaragua.

[15] Herrera, R. y. (2005). *Reconocimiento de comandos de voz para Windws y Word.* Recuperado el Febrero de 2013, de http://www.fip.unam.mx/simposio_investigacion2005/potencia29_ext.html

[16] HOLGÍN, German; PERÉZ, Sandra y OROZCO, Álvaro. "*Curso Básico de Labiew 6I*". Pereira: Universidad Tecnológica de Pereira, 2002, 265p

[17] IBARRA, D. (2009). *Sistema interactivo basado en voz para control de cargas y monitoreo de sensores de seguridad, orientado a discapacitados.* Tesis de Grado en Ingeniería en Electrónica y Telecomunicaciones, EPN. Quito 2009.

[18] Ingrid, K. A. (1998). *Automatic Speech Recognition with the Parallel Cascade Neural Network*. Tokyo Japan.

[19] Jesús Tomás Girónes. *"El Gran Libro de Android*", Primera Edición, Marcobo.S.A, Barcelona-España, 2011.

[20] Koon, R. (1998) *Aplicaciones de la informática para el desarrollo del pensamiento en alumnos con discapacidad*. Argentina

[21] Kornhauser, D. (4 de Marzo de 1999). *Diseno de un reconocedor de voz de palabras aisladas publico.* Recuperado el 15 de Febrero de 2013, de http:bq.unam.mx/-daniel/gvoice/tesis/node22.html

[22] MANUAL BÁSICO DE PROGRAMACIÓN EN LABIEW DE MARTERHACKS, 83p.

[23] Manuel Báez, Álvaro Borrego, Jorge Cordero, Luis Cruz, Miguel González, Francisco Hernández, David Palomero, José Rodríguez de Llera, Daniel Sanz, Mariam Saucedo, Pilar Torralbo, Álvaro Zapata. "*Introducción a Android"*, Madrid, 2011.

[24] NAUNCE. (s.f.). NAUNCE. Recuperado el 3 de Mayo de 2013, de www.naunce.com/Dragon

[25] *National Instruments*. Ni-IMAQ Users Manual. Austin. Texas. 2003. 112p.

[26] Ordonel Alquinga Edison Paul, Quillupangui Luje Gloria Alexandra. "*Diseño E Implmetacion De Unsistema De Clasificacion De Rosas Aplicando Vision Artificial Con Labview"*, Quito, 2011.

[27] Platero Carlos, "*Apuntes De Vision Artificial*". Dpto. Elctronica, Automatica, E Informatica Industrial. Archivo Pdf. Gnu Free Documentation License. España

[28] Pomares Baeza, Jorge. "*Manual de Arduino*", Universidad de Alicante, 2009.

[29] Rabiner, L. J.-H. (1998). *Speech Recognition by Machine*, Chap. 47 in the Digital Signal Processing Handbook, CRC Press, IEEE Press.

[30] Ramírez Hernández, Enrique. "*Desarrollo de Aplicaciones para Dispositivos con Sistemas Operativo Android*". Valencia, 2011.

[31] Regalado Pacheco, Juan Gregorio. "*Curso de supervivencia con Arduino*", 44 p.

[32] Ruiz Gutiérrez, José Manuel. "*Arduino + Ethernet Shield*", Enero-2013, 42p.

[33] Salvador Gómez Oliver. "*Curso de Programación Android*", Madrid, 2011.

[34] "*Sistema de Sensacion, Segmentacion, Reconocimiento y Clasificacion de Objetos",* Prof. M.Sc. Kryscia Daviana Ramírez Benavides.

[35]Slavinshy, J.(1999). Speaker Verification. Extraido el 20 de marzo de 2013 desde *http://www.owlnet.rice.edu/~elec301/Projects99/wrcocee/endpt.htm*

[36]"*Tecnicas De Segmentacion De Imágenes*". Marcos Martin, 2004.

[37] "*Tecnicas De Segmentación En El Procesamiento Digital De Imágenes*", Dra. Nora La Sena Polimino, Lic. Ulises Roman Concha.

*[38]*NUANCE COMMUNICATIONS, I. (2011). *Dragon Naturally Speaking, GUIA DE USUARIO, VERSIÓN 11.*

[39]Villasis Reyes Leonardo Javier, Ramos Cordero Hernán Andrés. "*Diseño y Construcción de un Sistema Automático de Control E Inspección De Botellas Selladas con Tapas Tipo Rosca, para Optimizar Tiempos en el Proceso de Empaquetado Utilizando Procesamiento Digital de Imágenes".* Latacunga, 2012.

REFERENCIAS ELECTRÓNICAS

[1] http://www.bhtavanza.com/es/ingenieria/ingenieriaihd/57-proyectos-domotica/

[2] http://es.wikipedia.org/

[3] http://www.csun.edu/~rd436460/Labview/IMAQ-Manual.pdf/

[4] *Monografias*. (s.f.). Recuperado el 6 de Mayo de 2013, de http://www.monografias.com/trabajos81/tutorial-programacion avanzada-visual-basic-6

[5] (s.f.). Obtenido de http://www.cepeu.edu.pyl/CURSO%20DE%20VISUAL%20%ESTUDIO%202008/Manual%20de%20Microsoft%20Visual%20Basic%202008.htm

[6] http://www.arduino.cc/

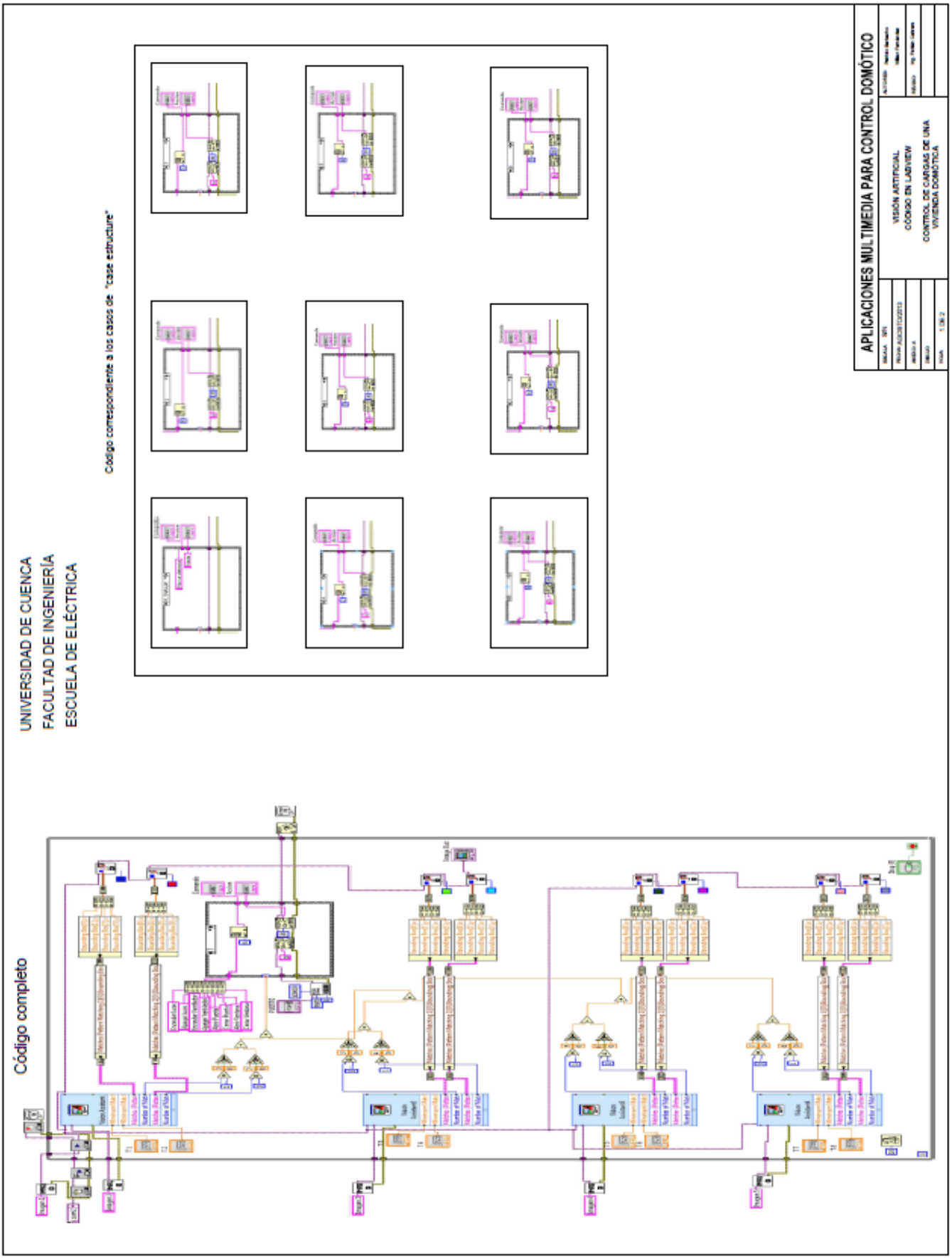
[7] http://androideity.com/

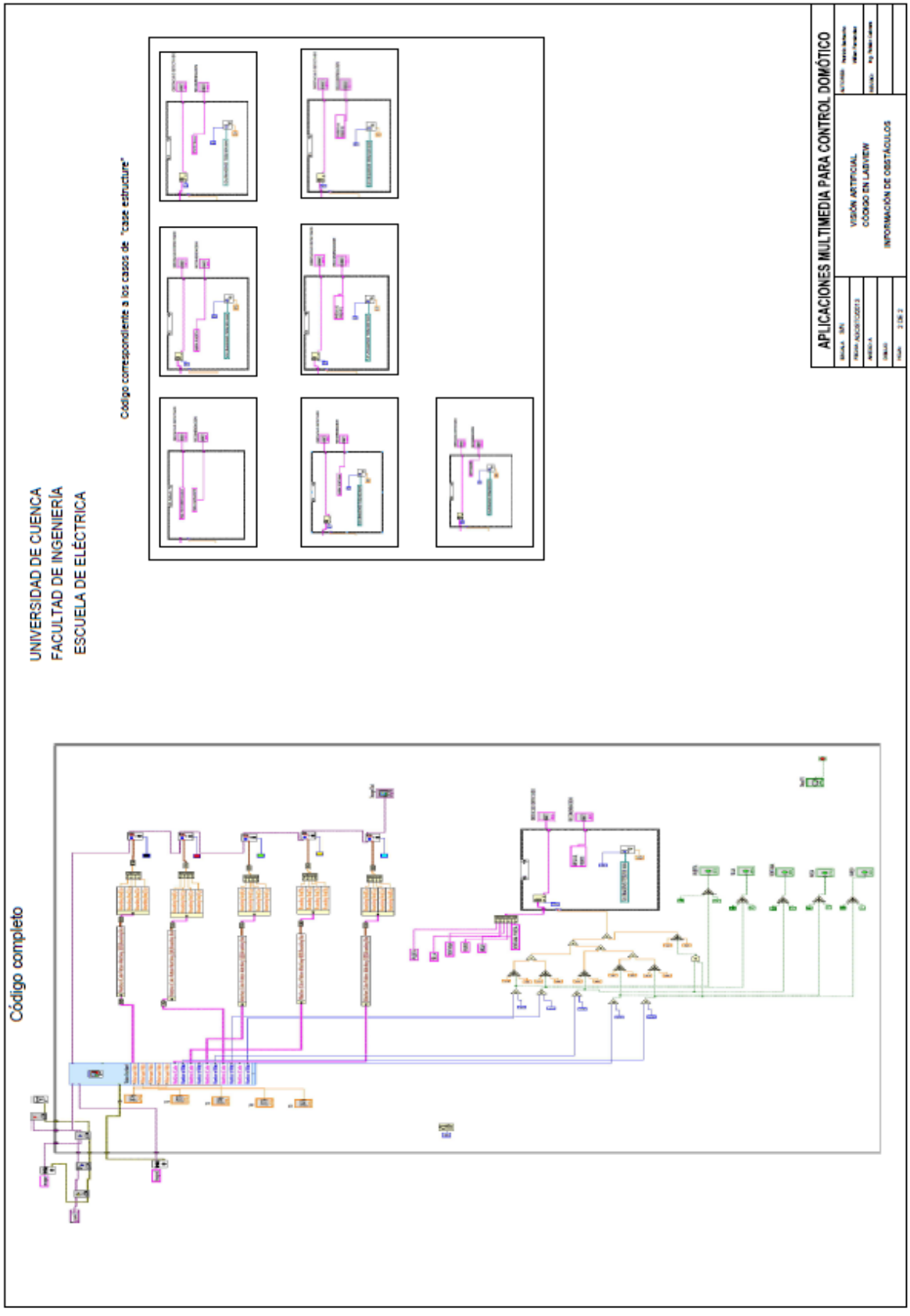
[8] http://developer.android.com/index.html

ANEXOS

ANEXO A

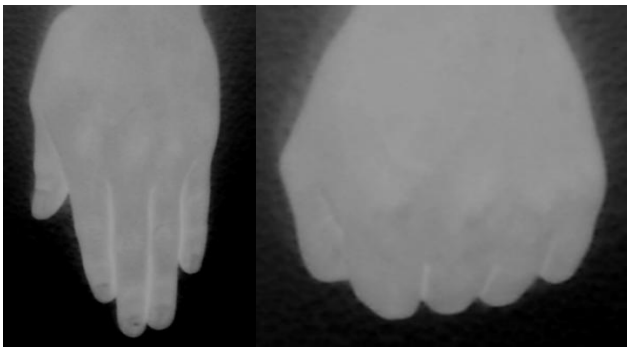
A.1 Código, para control domótico mediante visión artificial e información de obstáculos.

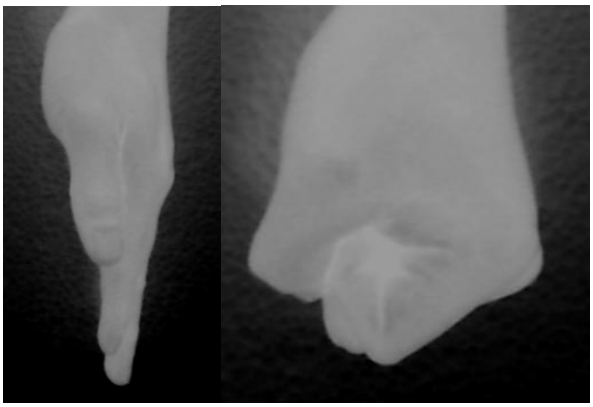

APLICACIONES MULTIMEDIA PARA CONTROL DOMÓTICO
VISIÓN ARTIFICIAL
CÓDIGO EN LABVIEW
CONTROL DE CARGAS DE UNA VIVIENDA DOMÓTICA
UNIVERSIDAD DE CUENCA
FACULTAD DE INGENIERÍA
ESCUELA DE ELÉCTRICA
Código correspondiente a los casos de "case estructure"
Código completo

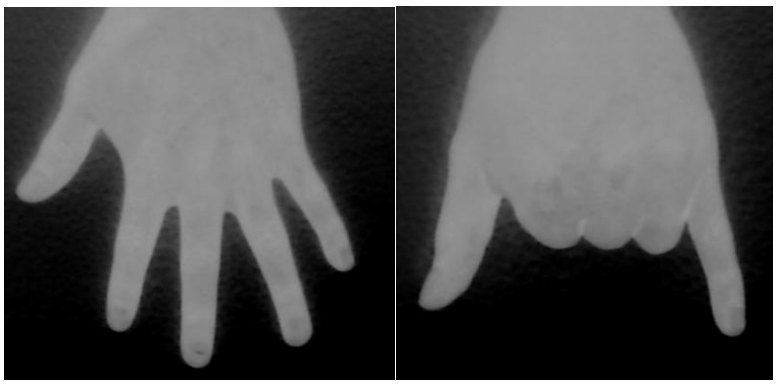

APLICACIONES MULTIMEDIA PARA CONTROL DOMÓTICO
VISIÓN ARTIFICIAL
CÓDIGO EN LABVIEW
INFORMACIÓN DE OBSTÁCULOS
UNIVERSIDAD DE CUENCA
FACULTAD DE INGENIERÍA
ESCUELA DE ELÉCTRICA
Código correspondiente a los casos de "case estructure"
Código completo

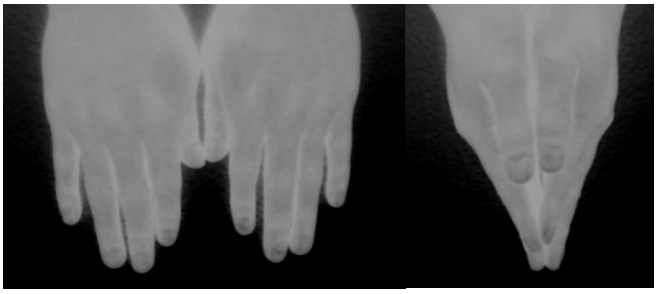
A.2 Templates utilizados en el sistema de visión artificial.

-Señas Para control de cargas

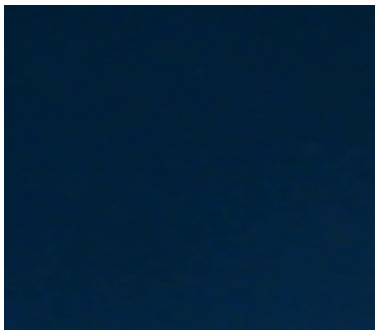








-**Colores para la señalización de obstáculos.**


ANEXO B

Código para control domótico mediante comandos de voz

B.1 CODIGO EN VISUAL BASIC 10

Definición de Variables

```
Dim Ivar1 As String

    Dim Ivar2 As String

    Dim Ivar3 As String

    Dim Svar1 As Integer

    Dim Pvar1 As Integer

    Dim as String

    Dim IBAN As String = "c"

    Dim Ban1 As String

    Dim Ban2 As String

    Dim Ban3 As String

    Dim Var_Puer As String

    Dim Aux As String

    Dim Aux1 As String

    Dim Coman1 As String
```

Comunicación serial

```
Private Sub VOZ_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

        GetSerialPortNames()

        CheckForIllegalCrossThreadCalls = False ' DESACTIVA ERROR POR
SUBPROCESO

    End Sub

    Sub Setup_Puerto_Serie()

        Try
```

```
            With SerialPort1


                If .IsOpen Then
                    .Close()
                End If


                .PortName = Cmb_puertos.Text         '// es el nombre del
combobox
                .BaudRate = 9600                    '// 19200 baud rate
                .DataBits = 8                        '// 8 data bits
                .StopBits = IO.Ports.StopBits.One    '// 1 Stop bit
                .Parity = IO.Ports.Parity.None       '
                .DtrEnable = False
                .Handshake = IO.Ports.Handshake.None
                .ReadBufferSize = 2048
                .WriteBufferSize = 1024
                '.ReceivedBytesThreshold = 1
                .WriteTimeout = 500
                .Encoding = System.Text.Encoding.Default
                .Open()                          '  ABRE EL PUERTO SERIE
            End With
        Catch ex As Exception
            MsgBox("Error al abrir el puerto serial: " & ex.Message,
MsgBoxStyle.Critical)
        End Try
    End Sub
    Sub GetSerialPortNames()
        ' muestra COM ports disponibles.
        Dim l As Integer
        Dim ncom As String
```

```
        Try
            Cmb_puertos.Items.Clear()
            For Each sp As String In My.Computer.Ports.SerialPortNames
                l = sp.Length
                If ((sp(l - 1) >= "0") And (sp(l - 1) <= "9")) Then
                    Cmb_puertos.Items.Add(sp)

                Else
                    'hay una letra al final del COM
                    ncom = sp.Substring(0, l - 1)
                    Cmb_puertos.Items.Add(ncom)
                End If
            Next
            If Cmb_puertos.Items.Count >= 1 Then
                Cmb_puertos.Text = Cmb_puertos.Items(0)
            Else
                Cmb_puertos.Text = ""
            End If

        Catch ex As Exception

        End Try
    End Sub
```

Código para control mediante comandos de voz

```
Private Sub Btt_inicio_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Btt_inicio.Click
        Try
            Setup_Puerto_Serie()
        Catch ex As Exception
        End Try
```

```
        Try

            Ptb_car.Image = New Bitmap("C:\Users\usuario\Desktop\Comandos
grabados\Imagenes\logo.png")

        Catch ex As Exception

            MessageBox.Show(ex.Message, ex.Source)

            Ptb_car.Image = Nothing

        End Try

        My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\40.wav", _

         AudioPlayMode.WaitToComplete)

        Txt_comando.Focus()


    End Sub


    Private Sub Txt_comando_TextChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Txt_comando.TextChanged

        Coman1 = Me.Txt_comando.Text


        If Coman1 = "dictar" Then

            Llb_titulo.Text = "CARGA A CONTROLAR"

            Label4.Text = "ILUMINACION"

            Label5.Text = "PUERTA"

            Label6.Text = "VENTANA"

            Label8.Text = "VENTILADOR"

            Label9.Text = "AYUDA"

            Me.Coman1 = ""

            Me.Txt_comando.Text = ""

            My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\41.wav", _

          AudioPlayMode.WaitToComplete)
```

```
            Me.Txt_comando.Text = ""
        End If



        If Coman1 = "iluminacion" Then
            IBAN = "i"
            Me.Coman1 = ""
            Me.Txt_comando.Text = ""
            My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\51.wav", _
         AudioPlayMode.Background)
        End If


        If Coman1 = "puerta" Then
            IBAN = "p"
            Me.Coman1 = ""
            Me.Txt_comando.Text = ""
            My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\52.wav", _
         AudioPlayMode.Background)
        End If


        If Coman1 = "ventana" Then
            IBAN = "v"
            Me.Coman1 = ""
            Me.Txt_comando.Text = ""
            My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\53.wav", _
         AudioPlayMode.Background)
        End If
        If Coman1 = "persianaa" Then
```

```
            IBAN = "pr"
            Me.Coman1 = ""
            Me.Txt_comando.Text = ""
            My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\1.wav", _
         AudioPlayMode.Background)
        End If
        If Coman1 = "ventilador" Then
            IBAN = "t"
            Me.Coman1 = ""
            Me.Txt_comando.Text = ""
            My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\54.wav", _
         AudioPlayMode.Background)
        End If
        If Coman1 = "emergencia" Then
            IBAN = "u"
            Me.Coman1 = ""
            Me.Txt_comando.Text = ""
        End If

        If Coman1 = "salir" Then
            Try
                If SerialPort1.IsOpen Then
                    SerialPort1.Close()
                End If
            Catch ex As Exception
            End Try
            Me.Txt_comando.Text = ""
            Me.Close()
```

```
        End If



        If IBAN = "i" Then
            If Coman1 = "estado actual" Then
                SerialPort1.Write("i")
                If Ivar1 = "encendido" Then
                    Me.Txt_Imens.Text = "DICTE COMANDO PARA APAGAR LUMINARIA"
                    Ivar1 = ""
                    Me.Coman1 = ""
                    Me.Txt_comando.Text = ""
                    My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\9.wav", _
         AudioPlayMode.Background)
                Else

                    Me.Txt_Imens.Text = "DICTE COMANDO PARA ENCENDER
LUMINARIA"
                    Me.Coman1 = ""
                    Me.Txt_comando.Text = ""
                    My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\8.wav", _
         AudioPlayMode.Background)
                End If
            End If



            If Coman1 = "apagar luminaria" Then
                Me.Txt_Imens.Text = "LUMINARIA APAGADA"
                Me.Coman1 = ""
                Me.Txt_comando.Text = ""
```

```
                Ivar1 = "apagado"

                Try

                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\focoapagado.jpg")

                Catch ex As Exception

                    MessageBox.Show(ex.Message, ex.Source)

                    Ptb_car.Image = Nothing

                End Try

                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\43.wav", _

         AudioPlayMode.Background)

                SerialPort1.Write("a")

            End If


            If Coman1 = "encender luminaria" Then

                Me.Txt_Imens.Text = "LUMINARIA ENCENDIDA"

                Me.Coman1 = ""

                Me.Txt_comando.Text = ""

                Ivar1 = "encendido"

                Try

                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos
grabados\Imagenes\focoencendido.jpg")

                Catch ex As Exception

                    MessageBox.Show(ex.Message, ex.Source)

                    Ptb_car.Image = Nothing

                End Try


                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\42.wav", _

         AudioPlayMode.Background)

                SerialPort1.Write("e")
```

```
            End If
            If Coman1 = "opciones" Then
                IBAN = "s"
                Me.Coman1 = ""
                Me.Txt_comando.Text = ""
                Me.Txt_Imens.Text = ""
                Try
                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\logo.png")
                Catch ex As Exception
                    MessageBox.Show(ex.Message, ex.Source)
                    Ptb_car.Image = Nothing
                End Try
                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\44.wav", _
        AudioPlayMode.Background)
            End If
        End If

        If IBAN = "p" Then
            If Coman1 = "estado actual" Then

                SerialPort1.Write("p")
                If Ivar1 = "abierto" Then
                    Me.Txt_Imens.Text = "DICTE COMANDO PARA CERRAR PUERTA"
                    Me.Coman1 = ""
                    Me.Txt_comando.Text = ""
                    My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\11.wav", _
        AudioPlayMode.Background)
```

```
                Else


                    Me.Txt_Imens.Text = "DICTE COMANDO PARA ABRIR PUERTA"

                    Me.Coman1 = ""

                    Me.Txt_comando.Text = ""

                    My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\10.wav", _
        AudioPlayMode.Background)


                End If

            End If

            If Coman1 = "abrir puerta" Then

                Me.Txt_Imens.Text = "ABRIENDO PUERTA"

                Me.Coman1 = ""

                Me.Txt_comando.Text = ""

                Ivar1 = "abierto"

                Try

                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos
grabados\Imagenes\puertaabierta.jpg")

                Catch ex As Exception

                    MessageBox.Show(ex.Message, ex.Source)

                    Ptb_car.Image = Nothing

                End Try

                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\2.wav", _
        AudioPlayMode.Background)

                SerialPort1.Write("b")

            End If


            If Coman1 = "cerrar puerta" Then
```

```
                Me.Txt_Imens.Text = "CERRANDO PUERTA"

                Me.Coman1 = ""

                Me.Txt_comando.Text = ""

                Ivar1 = "cerrrada"

                Try

                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos
grabados\Imagenes\puertacerrada.jpg")

                Catch ex As Exception

                    MessageBox.Show(ex.Message, ex.Source)

                    Ptb_car.Image = Nothing

                End Try

                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\3.wav", _

        AudioPlayMode.Background)

                SerialPort1.Write("c")

            End If

            If Coman1 = "opciones" Then

                IBAN = "s"

                Me.Coman1 = ""

                Me.Txt_comando.Text = ""

                Me.Txt_Imens.Text = ""

                Try

                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\logo.png")

                Catch ex As Exception

                    MessageBox.Show(ex.Message, ex.Source)

                    Ptb_car.Image = Nothing

                End Try

                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\44.wav", _

        AudioPlayMode.Background)
```

```
            End If
        End If
        If IBAN = "v" Then
            If Coman1 = "estado actual" Then

                SerialPort1.Write("v")
                If Ivar1 = "vabierto" Then
                    Me.Txt_Imens.Text = "DICTE COMANDO PARA CERRAR VENTANA"
                    Me.Coman1 = ""
                    Me.Txt_comando.Text = ""
                    My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\13.wav", _
        AudioPlayMode.Background)
                Else
                    Me.Txt_Imens.Text = "DICTE COMANDO PARA ABRIR VENTANA"
                    Me.Coman1 = ""
                    Me.Txt_comando.Text = ""
                    My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\12.wav", _
        AudioPlayMode.Background)

                End If
            End If
            If Coman1 = "abrir ventana" Then
                Me.Txt_Imens.Text = "ABRIENDO VENTANA"
                Me.Coman1 = ""
                Me.Txt_comando.Text = ""
                Ivar1 = "vabierto"
                Try
                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos
```

```
grabados\Imagenes\ventanaabierta.jpg")
                Catch ex As Exception
                    MessageBox.Show(ex.Message, ex.Source)
                    Ptb_car.Image = Nothing
                End Try
                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\4.wav", _
        AudioPlayMode.Background)
                SerialPort1.Write("d")
            End If

            If Coman1 = "cerrar ventana" Then
                Me.Txt_Imens.Text = "CERRANDO VENTANA"
                Me.Coman1 = ""
                Me.Txt_comando.Text = ""
                Ivar1 = "vcerrado"
                Try
                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos
grabados\Imagenes\ventanacerrada.jpg")
                Catch ex As Exception
                    MessageBox.Show(ex.Message, ex.Source)
                    Ptb_car.Image = Nothing
                End Try
                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\5.wav", _
        AudioPlayMode.Background)
                SerialPort1.Write("m")
            End If
            If Coman1 = "opciones" Then
                IBAN = "s"
```

```
                Me.Coman1 = ""

                Me.Txt_comando.Text = ""

                Me.Txt_Imens.Text = ""

                Try

                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\logo.png")

                Catch ex As Exception

                    MessageBox.Show(ex.Message, ex.Source)

                    Ptb_car.Image = Nothing

                End Try

                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\44.wav", _

        AudioPlayMode.Background)

            End If


        End If


        If IBAN = "pr" Then

            If Coman1 = "estado actual" Then


                SerialPort1.Write("z")

                If Ivar1 = "prabierto" Then

                    Me.Txt_Imens.Text = " DICTE COMANDO PARA CERRAR PERSIANA"

                    Me.Coman1 = ""

                    Me.Txt_comando.Text = ""

                    My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\27.wav", _

        AudioPlayMode.Background)

                Else

                    Me.Txt_Imens.Text = "DICTE COMANDO PARA ABRIR PERSIANA"

                    Me.Coman1 = ""
```

```
                    Me.Txt_comando.Text = ""
                    My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\26.wav", _
        AudioPlayMode.Background)


                End If
            End If
            If Coman1 = "abrir persiana" Then
                Me.Txt_Imens.Text = "PERSIANA ABIERTA"
                Me.Coman1 = ""
                Me.Txt_comando.Text = ""
                Try
                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\perabierta.jpg")
                Catch ex As Exception
                    MessageBox.Show(ex.Message, ex.Source)
                    Ptb_car.Image = Nothing
                End Try
                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\15.wav", _
        AudioPlayMode.Background)
                SerialPort1.Write("f")
            End If


            If Coman1 = "cerrar persiana" Then
                Me.Txt_Imens.Text = "PERSIANA CERRADA"
                Me.Coman1 = ""
                Me.Txt_comando.Text = ""
                Try
                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\percerrada.jpg")
```

```
                Catch ex As Exception
                    MessageBox.Show(ex.Message, ex.Source)
                    Ptb_car.Image = Nothing
                End Try
                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\16.wav", _
        AudioPlayMode.Background)
                SerialPort1.Write("g")
            End If
            If Coman1 = "opciones" Then
                IBAN = "s"
                Me.Coman1 = ""
                Me.Txt_comando.Text = ""
                Me.Txt_Imens.Text = ""
                Try
                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\logo.png")
                Catch ex As Exception
                    MessageBox.Show(ex.Message, ex.Source)
                    Ptb_car.Image = Nothing
                End Try
                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\44.wav", _
       AudioPlayMode.Background)
            End If


        End If


        If IBAN = "t" Then
            If Coman1 = "estado actual" Then
```

```
                SerialPort1.Write("x")
                If Ivar1 = "vactivo" Then
                    Me.Txt_Imens.Text = " DICTE COMANDO PARA DESACTIVAR
VENTILADOR"
                    Me.Coman1 = ""
                    Me.Txt_comando.Text = ""
                    My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\17.wav", _
        AudioPlayMode.Background)
                Else
                    Me.Txt_Imens.Text = "DICTE COMANDO PARA ACTIVAR
VENTILADOR"
                    Me.Coman1 = ""
                    Me.Txt_comando.Text = ""
                    My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\16.wav", _
        AudioPlayMode.Background)
                End If
            End If
            If Coman1 = "activar ventilador" Then
                Me.Txt_Imens.Text = "VENTILADOR ACTIVADO"
                Me.Coman1 = ""
                Me.Txt_comando.Text = ""
                Ivar1 = "vactivo"
                Try
                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\ventiladoren.jpg")
                Catch ex As Exception
                    MessageBox.Show(ex.Message, ex.Source)
                    Ptb_car.Image = Nothing
                End Try
                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
```

```
grabados\18.wav", _
        AudioPlayMode.Background)
                SerialPort1.Write("h")
            End If

            If Coman1 = "desactivar ventilador" Then
                Me.Txt_Imens.Text = "VENTILADOR DESACTIVADO"
                Me.Coman1 = ""
                Me.Txt_comando.Text = ""
                Ivar1 = "vdesactivo"
                Try
                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\ventiladorap.jpg")
                Catch ex As Exception
                    MessageBox.Show(ex.Message, ex.Source)
                    Ptb_car.Image = Nothing
                End Try
                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\19.wav", _
        AudioPlayMode.Background)
                SerialPort1.Write("n")
            End If
            If Coman1 = "opciones" Then
                IBAN = "s"
                Me.Coman1 = ""
                Me.Txt_comando.Text = ""
                Me.Txt_Imens.Text = ""
                Try
                    Ptb_car.Image = New
Bitmap("C:\Users\usuario\Desktop\Comandos grabados\Imagenes\logo.png")
                Catch ex As Exception
```

```
                    MessageBox.Show(ex.Message, ex.Source)
                    Ptb_car.Image = Nothing
                End Try
                My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\44.wav", _
       AudioPlayMode.Background)
            End If
        End If

        If IBAN = "u" Then
            IBAN = "s"
            SerialPort1.Write("y")
            Txt_Imens.Text = "ALARMA DE AYUDA ACTIVADA"
            Me.Coman1 = ""
            Me.Txt_comando.Text = ""
            Try
                Ptb_car.Image = New Bitmap("C:\Users\usuario\Desktop\Comandos
grabados\Imagenes\alarma.jpg")
            Catch ex As Exception
                MessageBox.Show(ex.Message, ex.Source)
                Ptb_car.Image = Nothing
            End Try
            My.Computer.Audio.Play("C:\Users\usuario\Desktop\Comandos
grabados\45.wav", _
         AudioPlayMode.Background)

        End If

    End Sub

    Private Sub SerialPort1_DataReceived(ByVal sender As Object, ByVal e As
```

```
System.IO.Ports.SerialDataReceivedEventArgs) Handles SerialPort1.DataReceived
        Try
            Dim s As String
            az = SerialPort1.ReadExisting.Trim
            s = az
            If s = "E" Then



            End If

            If s = "A" Then


            End If
            If s = "B" Then
                Ivar1 = "abierto"
            End If
            If s = "C" Then
                Ivar1 = ""
            End If
            If s = "D" Then
                Ivar1 = "vabierto"
            End If
            If s = "M" Then
                Ivar1 = ""
            End If
            If s = "F" Then
                Ivar1 = "prabierto"
            End If
            If s = "G" Then
```

```
                Ivar1 = "vactivo"
            End If
            If s = "H" Then
                Ivar1 = ""
            End If
            If s = "N" Then
                Ivar1 = ""
            End If
            If s = "j" Then

                Me.Txt_Imens.Text = " ALARMA ACTIVADA"

            End If

        Catch ex As Exception
            MsgBox(ex.Message)
        End Try

    End Sub

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer1.Tick
        Label2.Text = TimeString
        Me.Label3.Text = Date.Today
    End Sub
End Class
```

B.2 CODIGO EN PROTON IDE PARA CONTROL DOMOTICO MEDIANTE COMANDOS DE VOZ.

Código para control mediante comandos de voz.

```
'****************************************************************
'*  Name    : Comandos de voz.BAS                                *
'*  Author  : Patricio Barbecho y Willian Fernández              *
'*  Notice  : Copyright (c) 2013 [select VIEW...EDITOR OPTIONS] *
'*          : All Rights Reserved                                *
'*  Date    : 4/30/2013                                          *
'*  Version : 1.0                                                *
'*  Notes   : Programa para control on-off de cargas             *
'*          : y comunicación a través del puerto serial.         *
'****************************************************************

     Device 16F877A
XTAL=4


Declare I2C_BUS_SCL On    ;para que no necesite resistencia pull-up
en SCL
Include "modedefs.bas"


Dim band  As  Byte
Dim dato3 As Byte
Dim dato4 As Byte
Dim i As Byte
Dim Svar1 As Byte
Dim Svar2 As Byte
```

```
Dim Svar3 As Byte
Dim Vvar1 As Byte
Dim Ban1 As Byte
Dim Eluminaria As Byte
Dim Epuerta As Byte
Dim Eventana As Byte
Dim Eventilador As Byte

TRISC.7=1
TRISC.6=0

ALL_DIGITAL=true

;CMCON=7 ;Todo el puerto A en modo digital
PORTA=1
PORTB=0

Eluminaria="a"
Epuerta= "c"
Eventana= "c"
Eventilador="a"

Dim Servo_Posicion As Word

DelayMS 150
Cls
Clear
TRISB=0
 PORTB.1=0
```

```
 PORTB.2=0
Servo_Posicion = 1500
 Servo PORTB.1, Servo_Posicion
 Servo PORTB.2, Servo_Posicion

principal:

  SerIn PORTC.7,N9600,500,pato,[dato3]
  If dato3="i" Then
  GoSub Iluminacion
  EndIf
  If dato3="e" Then
  GoSub Sencendido
  End If
  If dato3="a" Then
  GoSub Sapagado
  End If
  If dato3="p" Then
  GoSub Puerta
  End If
  If dato3="b" Then
  GoSub pactivar
  End If
  If dato3="c" Then
  GoSub pdesactivar
  End If
  If dato3="v" Then
  GoSub Ventana
  End If
```

```
  If dato3="d" Then
  GoSub vactivar
  End If
  If dato3="m" Then
  GoSub vdesactivar
  End If

  If dato3="x" Then
  GoSub Ventilador
  End If
  If dato3="h" Then
  GoSub vnactivar
  End If
  If dato3="n" Then
  GoSub vndesactivar
  End If
  If dato3="y" Then
  GoSub alractivar
  End If

GoTo principal

 pato:
 GoTo principal

 Iluminacion:
 GoSub ietd
 If Svar2="E" Then
 SerOut PORTC.6,N9600,["E"]
```

```
DelayMS 500
Else
SerOut PORTC.6,N9600,["A"]
DelayMS 500
End If
Return

Puerta:
GoSub petd
If Svar2="B" Then
SerOut PORTC.6,N9600,["B"]
DelayMS 500
Else
SerOut PORTC.6,N9600,["C"]
DelayMS 500
End If
Return

ietd:
If Eluminaria="a" Then
Svar2="A"
End If
If Eluminaria="e" Then
Svar2="E"
End If
Return

 petd:
If Epuerta="c" Then
```

```
Svar2="C"
End If
If Epuerta="a" Then
Svar2="B"
End If
Return

Sencendido:
PORTB.0 = 1
Eluminaria="e"
SerOut PORTC.6,N9600,["e"]
DelayMS 500
Return

Sapagado:
PORTB.0 = 0
Eluminaria="a"
SerOut PORTC.6,N9600,["a"]
DelayMS 500
Return

 pdesactivar:
 Epuerta="c"
Repeat
Servo PORTB.1, Servo_Posicion
DelayMS 10
Inc Servo_Posicion
Until Servo_Posicion = 1500
SerOut PORTC.6,N9600,["c"]
```

```
DelayMS 500
Return

pactivar:
Epuerta="a"
Repeat
Servo PORTB.1, Servo_Posicion
DelayMS 10
Dec Servo_Posicion
Until Servo_Posicion = 500
SerOut PORTC.6,N9600,["b"]
DelayMS 500
Return

Ventana:
GoSub vetd
If Svar2="D" Then
SerOut PORTC.6,N9600,["D"]
DelayMS 500
Else
SerOut PORTC.6,N9600,["M"]
DelayMS 500
End If
Return

vetd:
If Eventana="c" Then
Svar2="M"
End If
```

```
If Eventana="a" Then
Svar2="D"
End If
Return

vdesactivar:
Eventana="m"
 Repeat
Servo PORTB.2, Servo_Posicion
DelayMS 10
Inc Servo_Posicion
Until Servo_Posicion = 1500
SerOut PORTC.6,N9600,["m"]
DelayMS 500
Return

vactivar:
Eventana="a"
Repeat
Servo PORTB.2, Servo_Posicion
DelayMS 10
Dec Servo_Posicion
Until Servo_Posicion = 500
SerOut PORTC.6,N9600,["d"]
DelayMS 500
Return

Ventilador:
GoSub vnetd
```

```
If Svar2="H" Then
SerOut PORTC.6,N9600,["H"]
DelayMS 500
Else
SerOut PORTC.6,N9600,["N"]
DelayMS 500
End If
Return

vnetd:
If Eventilador= "a" Then
Svar2="N"
End If
If Eventilador= "e" Then
Svar2="H"
End If
Return

vndesactivar:
PORTB.5 = 0
Eventilador= "a"
SerOut PORTC.6,N9600,["n"]
DelayMS 500
Return

vnactivar:
PORTB.5 = 1
Eventilador= "e"
SerOut PORTC.6,N9600,["h"]
```

```
DelayMS 500
Return

alractivar:
PORTB.6 = 1
DelayMS 3000
PORTB.6= 0
SerOut PORTC.6,N9600,["j"]
DelayMS 500
Return
```

Anexo C

APLICACIÓN ANDROID

A continuación se anexa todo el código java desarrollado en esta aplicación. Y los archivos xml que corresponden a las interfaces usuario.

C.1 Domotica (Main Activity)

```
package com.domotica;

import android.media.MediaPlayer;
import android.os.Bundle;
import android.view.KeyEvent;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.app.AlertDialog;
import android.app.ListActivity;
import android.content.DialogInterface;
import android.content.Intent;

public class Domotica extends ListActivity {
	MediaPlayer med;
	String []controles={"Vivienda","VideoVigilancia","Seguridad"};

	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setListAdapter(new
ArrayAdapter<String>(this,android.R.layout.simple_list_item_1,contro
les));

	}


	protected void onListItemClick(ListView list,View v,int
position,long id){
		super.onListItemClick(list, v, position, id);
		String nombreControl=controles[position];
		try {
			Class<?>
clas=Class.forName("com.domotica."+nombreControl);
			Intent intent=new Intent(this,clas);
			startActivity(intent);
		} catch (ClassNotFoundException e) {
			e.getStackTrace();
		}

	}
```

```
public void onStart(){
      super.onStart();
      med=MediaPlayer.create(this,R.raw.oprincipales);
      med.start();
}

public void onPause(){
      super.onPause();
      med.stop();
}

public void onStop(){
super.onStop();
med.stop();
med.release();
}

@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
      super.onKeyDown(keyCode, event);
      switch (keyCode) {
      case KeyEvent.KEYCODE_BACK:
            AlertDialog.Builder alerta=new
AlertDialog.Builder(this);
            alerta.setMessage("¿Estas seguro de
salir?").setCancelable(false).
            setPositiveButton("OK", new
DialogInterface.OnClickListener() {

                  @Override
                  public void onClick(DialogInterface dialog, int
id) {
                        onPause();
                        System.exit(RESULT_OK);

                  }
            }).setNegativeButton("Cancelar",new
DialogInterface.OnClickListener() {

                  @Override
                  public void onClick(DialogInterface dialog, int
id) {
                        dialog.cancel();

                  }
            } );
            alerta.create().show();

            break;

      }
      return true;


 }


}
```

C.2 Vivienda (Activity)

```
package com.domotica;



import android.media.MediaPlayer;
import android.os.Bundle;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.app.Activity;


public class Vivienda extends Activity {
	MediaPlayer med;
	WebView web_vivienda;

	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.control_domotico);
		web_vivienda=(WebView)findViewById(R.id.wb_control);

		web_vivienda.loadUrl("http://192.168.0.2/");
		web_vivienda.setWebViewClient(new ViviendaViewClient());
		web_vivienda.getSettings().setBuiltInZoomControls(true);
		web_vivienda.getSettings().setUseWideViewPort(true);
		web_vivienda.getSettings().setPluginsEnabled(true);

	}


	public void onStart(){
		super.onStart();
		med=MediaPlayer.create(this,R.raw.cdomotico);
		med.start();
	}

	public void onPause(){
		super.onPause();
		med.stop();
	}

	public void onStop(){
	super.onStop();
	med.stop();
	med.release();
	}

}
class ViviendaViewClient extends WebViewClient{

	public boolean shouldOverrideUrlLading(WebView v,String url){
		v.loadUrl(url);
		return true;

	}

}
```

C.2.1 Interfaz usuario control_domotico.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <WebView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/wb_control"
        android:focusable="true"/>
</LinearLayout>
```

C.3 VideoVigilancia (Activity)

```
package com.domotica;

import android.media.MediaPlayer;
import android.os.Bundle;
import android.app.Activity;
import android.view.View;
import android.view.Window;
import android.view.WindowManager;
import android.view.View.OnClickListener;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemSelectedListener;
import android.widget.ArrayAdapter;
import android.widget.ImageButton;
import android.widget.Spinner;
import android.widget.Toast;

public class VideoVigilancia extends Activity implements
OnClickListener,OnItemSelectedListener{
      MediaPlayer med;
      ImageButton play;
      WebView sitioweb;
      Thread actualizar;
      String url;
      Spinner s1;
      String[]camaras={"CAM1","CAM2","CAM3"};


      @Override
      protected void onCreate(Bundle savedInstanceState) {
            requestWindowFeature(Window.FEATURE_NO_TITLE);
```

```
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
                             WindowManager.LayoutParams.FLAG_FULLSCREEN);
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_video_vigilancia);



                sitioweb=(WebView)findViewById(R.id.vwNavegador);
                s1=(Spinner)findViewById(R.id.spinner1);

                ArrayAdapter<String> adapter=new ArrayAdapter<String>(this,

        android.R.layout.simple_spinner_dropdown_item,camaras);
                s1.setAdapter(adapter);
                s1.setOnItemSelectedListener(this);


                play=(ImageButton)findViewById(R.id.btplay);
                play.setOnClickListener(this);

                sitioweb.setWebViewClient(new ViewClient());//paraque no se salga de la aplicacion
                sitioweb.getSettings().setBuiltInZoomControls(true);
                sitioweb.getSettings().setUseWideViewPort(true);//para que se adapten en la WiebView
                sitioweb.getSettings().setPluginsEnabled(true);

                 actualizar=new Thread(){

                            @Override
                            public void run() {

                                    try {
                                          while(true){
                                          sleep(200);
                                          sitioweb.reload();
                                          }

                                    } catch (InterruptedException e)
{

        Toast.makeText(getBaseContext(),"se ha producido un error",Toast.LENGTH_LONG).show();
                                    }

                                }

                        };
        }

        public void onStart(){
                super.onStart();
                med=MediaPlayer.create(this,R.raw.vvigilancia);
                med.start();
        }

        public void onPause(){
```

```
            super.onPause();
            med.stop();
      }

      public void onStop(){
      super.onStop();
      med.stop();
      med.release();
      }


      @Override
      public void onClick(View v) {
            if (v==play){
                  try{
                        sitioweb.loadUrl(url);
                        actualizar.start();
                  }catch(Exception e){
                        Toast.makeText(this,"se ha perdido
conexion",Toast.LENGTH_LONG).show();
                        finish();

                  }

            }

      }

      @Override
      public void onItemSelected(AdapterView<?> adapter, View v, int
arg2,
                  long arg3) {
            int index=adapter.getSelectedItemPosition();
            switch (index) {
            case 0:
                  url="http://192.168.0.20/image/jpeg.cgi";
                  sitioweb.loadUrl(url);
                  break;

            case 1:
                  url="http://192.168.0.20/image/jpeg.cgi";
                  sitioweb.loadUrl(url);
                  Toast.makeText(this,"no existe camara 2
instalada",Toast.LENGTH_SHORT).show();
                  break;
            case 2:
                  url="http://192.168.0.20/image/jpeg.cgi";
                  sitioweb.loadUrl(url);
                  Toast.makeText(this,"no existe camara 3
instalada",Toast.LENGTH_SHORT).show();
                  break;
            }

      }

      @Override
      public void onNothingSelected(AdapterView<?> arg0) {
            // TODO Auto-generated method stub

      }
```

```
}

class ViewClient extends WebViewClient{

      public boolean shouldOverrideUrlLading(WebView v,String url){
            v.loadUrl(url);
            return true;

      }

}
```

C.3.1 Interfaz usuario activity_video_vigilancia.xml

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".VideoVigilancia" >

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="53dp"
        android:orientation="horizontal"
        android:weightSum="10" >

        <Spinner
            android:id="@+id/spinner1"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="4"
/>

        <ImageButton
            android:id="@+id/btplay"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="6"
            android:src="@drawable/play"
            />


    </LinearLayout>

    <WebView
        android:id="@+id/vwNavegador"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:focusable="true"/>

</LinearLayout>
```

C.4 Seguridad (Activity)

```
package com.domotica;

import android.media.MediaPlayer;
import android.os.Bundle;
import android.app.Activity;
import android.app.AlertDialog;
import android.app.Dialog;
import android.app.PendingIntent;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.CursorIndexOutOfBoundsException;
import android.telephony.SmsManager;
import android.telephony.SmsMessage;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

public class Seguridad extends Activity implements OnClickListener {
MediaPlayer med;
Button btEnviarSms;
AlertDialog.Builder alerta;



	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_seguridad);

		btEnviarSms=(Button)findViewById(R.id.btenviarSms);
		btEnviarSms.setOnClickListener(this);

	}

	public void onStart(){
		super.onStart();
		med=MediaPlayer.create(this,R.raw.emergencia);
		med.start();
	}

	public void onPause(){
		super.onPause();
		med.stop();
	}

	public void onStop(){
	super.onStop();
	med.stop();
	med.release();
	}


	@Override
	public void onClick(View v) {

		if(v==btEnviarSms){
```

```
                    alerta=new AlertDialog.Builder(this);
                    alerta.setMessage("¿Estas seguro de enviar los
mensajes?").setCancelable(false)
                    .setPositiveButton("SI", new
DialogInterface.OnClickListener() {

                        @Override
                        public void onClick(DialogInterface dialog,
int id) {
                            SQLiteSMS baseDatos=new
SQLiteSMS(getBaseContext());
                            try {
                                baseDatos.abrir();
                            } catch (Exception e) {
                                crearDialogo("Error al abrir base
de datos");
                            }
                            try{
                          for(int i=1;i<6;i++) {

              String mensString=baseDatos.getMensaje(i);
                            String
telString=baseDatos.getTelefono(i);

                            if(!mensString.equals("NE")){
                              enviarSms(telString,mensString);
                              String text="MENSAJE ENVIADO...\n"+"ID:
"+i+"\nPara: "+telString+"\n"+

      "SMS:"+mensString+"\n"+

      "\n¡Operación realizada Exitosamente!";
                                                    Toast
toast=Toast.makeText(getBaseContext(),text, Toast.LENGTH_LONG);
                                                    toast.show();
                            }


                            }
                              baseDatos.cerrar();

                            }catch(CursorIndexOutOfBoundsException
e){
                              crearDialogo("No existen datos
guardados");

                            }catch(NullPointerException e){
                              crearDialogo("Fatal Error");

                            }catch (Exception e) {
                                crearDialogo("No se puede
realizar la operacion");
                            }

                        }
                    }).setNegativeButton("No", new
DialogInterface.OnClickListener() {

                        @Override
```

```
                        public void onClick(DialogInterface dialog,
int id) {
                              dialog.cancel();

                        }
                  });
                  alerta.create();
                  alerta.show();


            }


            }

      public void enviarSms(String number,String text){
            PendingIntent pi=PendingIntent.getActivity(this,0,new
Intent(this,SmsMessage.class),0);
            SmsManager sms=SmsManager.getDefault();
            sms.sendTextMessage(number, null, text, pi, null);

}
      public void crearDialogo(String error) {
            Dialog d=new Dialog(this);
            d.setTitle("Error");
            TextView tv=new TextView(this);
            tv.setTextColor(0xFFFFFFFF);
            tv.setText(error);
            d.setContentView(tv);
            d.show();
      }

      @Override
      public boolean onCreateOptionsMenu(Menu menu) {
            getMenuInflater().inflate(R.menu.seguridad, menu);
            return true;
      }

      @Override
      public boolean onOptionsItemSelected(MenuItem item) {
            switch (item.getItemId()) {

            case R.id.opcionesMensajes:
                  Intent i=new Intent(this, Configuraciones.class);
                  startActivity(i);
                  return true;
            case R.id.ayudaSMS:
                  String text="Al presionar sobre el botón se
enviara mensajes " +
                              "con el texto que tu decidas. Para
insertar tus contactos " +
                              "y el SMS a enviar ve a
configuraciones.";
                              Dialog d=new Dialog(this);
                              d.setTitle("¡Ayuda!");
                              TextView tv=new TextView(this);
                              tv.setText(text);
                              tv.setTextColor(0xFFFFFFFF);
                              d.setContentView(tv);
                              d.show();
```

```
                return true;

            default:
                return super.onOptionsItemSelected(item);
            }


        }

        }
```

C.4.1 Interfaz usuario activity_seguridad.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FF4400"
    android:id="@+id/rl1" >

    <Button
        android:id="@+id/btenviarSms"
        android:layout_width="130dp"
        android:layout_height="120dp"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:background="@layout/boton" />

    <TextView
        android:id="@+id/tv"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/btenviarSms"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="32dp"
        android:text="Emergencia"
        android:textSize="40sp" />

</RelativeLayout>
```

C.5 SQLiteSMS (Clase)

```
package com.domotica;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.CursorIndexOutOfBoundsException;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class SQLiteSMS {
      public static final String ID_FILA="_id";
      public static final String ID_MENSAJE="mensaje";
      public static final String ID_TELEFONO="numero_telefono";

      private static final String N_BD="SQLiteSMS";
```

```
	private static final String N_TABLA="Tabla_Mensajes";
	private static final int VERSION_BD=1;

	private BDHelper nHelper;
	private final Context nContexto;
	private SQLiteDatabase nBD;

	public SQLiteSMS(Context c){
		nContexto =c;//constructor de SQliteSms
		nHelper=new BDHelper(nContexto);
	}

	private static class BDHelper extends SQLiteOpenHelper{

		public BDHelper(Context context){
			super(context,N_BD,null,VERSION_BD);

		}

		@Override
		public void onCreate(SQLiteDatabase db) {
			db.execSQL("CREATE TABLE " + N_TABLA + "(" +
					ID_FILA +" INTEGER PRIMARY KEY
AUTOINCREMENT, " +
					ID_MENSAJE +" TEXT NOT NULL, "+
					ID_TELEFONO +" TEXT NOT NULL);"
					);

		}

		@Override
		public void onUpgrade(SQLiteDatabase db, int oldVersion,
int newVersion) {
			db.execSQL("DROP TABLE IF EXIST "+N_TABLA);
			onCreate(db);

		}

	}



	public SQLiteSMS abrir()throws Exception{
		nHelper=new BDHelper(nContexto);
		nBD=nHelper.getWritableDatabase();
		return this;
	}

	public void cerrar() {
		nHelper.close();
	}

	public long crearDatos(String mensaje, String telefono) {
		ContentValues cv=new ContentValues();
		cv.put(ID_MENSAJE,mensaje);
		cv.put(ID_TELEFONO,telefono);
		return nBD.insert(N_TABLA,null,cv);//nos devuelve el id
de la fila
	}
```

```
	public String mostrarDatos() {

		String []columnas= new
String[]{ID_FILA,ID_MENSAJE,ID_TELEFONO};
		Cursor c=nBD.query(N_TABLA,
columnas,null,null,null,null,null);
		String resultado="";

		int iFila=c.getColumnIndex(ID_FILA);
		int iMensaje=c.getColumnIndex(ID_MENSAJE);
		int iTelefono=c.getColumnIndex(ID_TELEFONO);

		for(c.moveToFirst();!c.isAfterLast();c.moveToNext()){
			resultado=resultado+c.getString(iFila)+"
"+c.getString(iMensaje)+" "+c.getString(iTelefono)+"\n";

		}
		return resultado;
	}

	public String getMensaje(long numFila)throws
CursorIndexOutOfBoundsException {
		String []columnas= new
String[]{ID_FILA,ID_MENSAJE,ID_TELEFONO};
		Cursor c=nBD.query(N_TABLA,
columnas,ID_FILA+"="+numFila,null,null,null,null);
		if(c!=null){
			c.moveToFirst();
			String smsBuscado=c.getString(1);
			return smsBuscado;

		}
		return null;
	}


	public String getTelefono(long numFila)throws
CursorIndexOutOfBoundsException {
		// TODO Auto-generated method stub
		String []columnas= new
String[]{ID_FILA,ID_MENSAJE,ID_TELEFONO};
		Cursor c=nBD.query(N_TABLA,
columnas,ID_FILA+"="+numFila,null,null,null,null);
		if(c!=null){
			c.moveToFirst();
			String telefonoBuscado=c.getString(2);
			return telefonoBuscado;

		}
		return null;
	}

	public void guadar(long numFila, String mensaje, String
telefono)throws SQLException {
	//sirve para salvar la actualizacion de datos
		ContentValues cvEditar=new ContentValues();
		 cvEditar.put(ID_MENSAJE, mensaje);
		 cvEditar.put(ID_TELEFONO,telefono);
		 nBD.update(N_TABLA,cvEditar,ID_FILA+"="+numFila,null);
```

```
}

	public void borrar(long numFila)throws SQLException {

		nBD.delete(N_TABLA,ID_FILA+"="+numFila,null);
	}
}
```

C.6 Configuraciones (Activity)

```
package com.domotica;


import android.os.Bundle;
import android.app.Activity;
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.CursorIndexOutOfBoundsException;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class Configuraciones extends Activity implements
OnClickListener{
	EditText mensaje,telefono,edbuscar;
	Button insertar,ver,buscar,guardar,eliminar;

	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_configuraciones);
		mensaje=(EditText)findViewById(R.id.edMensaje);
		telefono=(EditText)findViewById(R.id.edTelefono);
		edbuscar=(EditText)findViewById(R.id.etBuscar);
		insertar=(Button)findViewById(R.id.btInsertar);
		ver=(Button)findViewById(R.id.btVer);
		buscar=(Button)findViewById(R.id.btBuscar);
		guardar=(Button)findViewById(R.id.btguardar);
		eliminar=(Button)findViewById(R.id.btEliminar);

		insertar.setOnClickListener(this);
		ver.setOnClickListener(this);
		buscar.setOnClickListener(this);
		guardar.setOnClickListener(this);
		eliminar.setOnClickListener(this);


	}
	@Override
	public void onClick(View v) {
		switch (v.getId()) {
		case R.id.btInsertar:
```

```
                String sms=mensaje.getText().toString();
                String tel=telefono.getText().toString();

                SQLiteSMS insertar=new SQLiteSMS(this);

                try {
                      insertar.abrir();
                } catch (Exception e) {
                      crearDialogo("Error al leer la base de
datos");
                }
                insertar.crearDatos(sms, tel);
                insertar.cerrar();
                borrarETexts();

                break;
          case R.id.btVer:
                Intent intent=new Intent(this,VerSQL.class);
                startActivity(intent);
                borrarETexts();


                break;
          case R.id.btBuscar:
                String idFila=edbuscar.getText().toString();
                long fila=1;
                try{
                 fila=Long.parseLong(idFila);
                }catch (NumberFormatException e) {
                      crearDialogo("Revise el inidador de Filas");
                      edbuscar.setText("1");
                }
                SQLiteSMS buscar=new SQLiteSMS(this);

                try {
                      buscar.abrir();
                } catch (Exception e) {
                      crearDialogo("Error al leer la base de
datos");
                }
                try{
                String men=buscar.getMensaje(fila);
                String tele=buscar.getTelefono(fila);
                buscar.cerrar();
                mensaje.setText(men);
                telefono.setText(tele);

                }catch (CursorIndexOutOfBoundsException e) {
                      crearDialogo("ERROR AL BUSCAR LOS DATOS POR
FAVOR REVISE EL INDICADOR DE FILAS");
                }



                break;
          case R.id.btguardar:

                AlertDialog.Builder alerta=new
AlertDialog.Builder(this);
```

```
                alerta.setMessage("¿Estas seguro de Guardar los
cambios?").setCancelable(false)
                .setPositiveButton("SI", new
DialogInterface.OnClickListener() {

                        @Override
                        public void onClick(DialogInterface dialog,
int id) {
                                String
edMen=mensaje.getText().toString();
                                String
edTel=telefono.getText().toString();
                                String
edFila=edbuscar.getText().toString();
                                long numFila=1;

                                SQLiteSMS actualizar=new
SQLiteSMS(getBaseContext());
                                try {
                                      actualizar.abrir();
                                } catch (Exception e) {
                                crearDialogo("Error al leer la base de
datos");
                                }
                                try{
                                numFila=Long.parseLong(edFila);
                                actualizar.guadar(numFila, edMen,
edTel);
                                actualizar.cerrar();
                                Toast.makeText(getBaseContext(), "¡La
operación se ha realizado exitosamente!", Toast.LENGTH_LONG).show();
                                borrarETexts();
                                }catch (Exception e) {
                                      crearDialogo("Revise el inidador
de Filas");
                                }

                        }
                }).setNegativeButton("No", new
DialogInterface.OnClickListener() {

                        @Override
                        public void onClick(DialogInterface dialog,
int id) {
                                dialog.cancel();

                        }
                });
                alerta.create();
                alerta.show();




                break;
          case R.id.btEliminar:
                alerta=new AlertDialog.Builder(this);
                alerta.setMessage("¿Estas seguro de eliminar estos
datos?").setCancelable(false)
```

```
                    .setPositiveButton("SI", new
DialogInterface.OnClickListener() {

                        @Override
                        public void onClick(DialogInterface dialog,
int id) {
                              String
elFila=edbuscar.getText().toString();
                              SQLiteSMS borrar=new
SQLiteSMS(getBaseContext());
                              long elFila1=0;
                              try {
                                    borrar.abrir();
                              } catch (Exception e) {
                                    crearDialogo("Error al leer la
base de datos");
                              }

                              try {
                                    elFila1=Long.parseLong(elFila);
                                    borrar.borrar(elFila1);
                                    borrar.cerrar();
                                    Toast.makeText(getBaseContext(),
"¡La operación se ha realizado exitosamente!",
Toast.LENGTH_LONG).show();
                                    borrarETexts();
                              } catch (Exception e) {
                                    crearDialogo("Revise el inidador
de Filas");
                              }


                        }
                    }).setNegativeButton("No", new
DialogInterface.OnClickListener() {

                        @Override
                        public void onClick(DialogInterface dialog,
int id) {
                              dialog.cancel();

                        }
                    });
                    alerta.create();
                    alerta.show();



                    break;


        }


      }
      public void borrarETexts() {
            mensaje.setText("");
            telefono.setText("");
            edbuscar.setText("");
```

```
	}
	public void crearDialogo(String error) {
		Dialog d=new Dialog(this);
		d.setTitle("Error");
		TextView tv=new TextView(this);
		tv.setText(error);
		tv.setTextColor(0xFFFFFFFF);
		d.setContentView(tv);
		d.show();
	}

	@Override
	public boolean onCreateOptionsMenu(Menu menu) {
		getMenuInflater().inflate(R.menu.configuraciones, menu);
		return true;
	}

	@Override
	public boolean onOptionsItemSelected(MenuItem item) {
		switch (item.getItemId()) {

		case R.id.ayudaConfiguraciones:
			String text="Inserta los mensajes con sus números
telefónicos con "+
			"un maximo de 5 sms. Para remplazar buscalos con su
id en lo posible no elimines datos "+
					"dado a que no se auto estructura.";
			Dialog d=new Dialog(this);
			d.setTitle("¡Ayuda!");
			TextView tv=new TextView(this);
			tv.setText(text);
			tv.setTextColor(0xFFFFFFFF);
			d.setContentView(tv);
			d.show();
			return true;
		default:
			return super.onOptionsItemSelected(item);
		}
	}

}
```

C.6.1 Interfaz usuario activity_configuraciones.xml

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:background="#000FFF">

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/edMensaje"
        android:hint="Ingrese un mensaje "/>
    <EditText
```

```
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/edTelefono"
        android:hint="Ingrese el Telefono"
        android:numeric="integer"/>

    <LinearLayout
       android:layout_width="fill_parent"
       android:layout_height="wrap_content"
       android:orientation="horizontal"
       android:weightSum="10" >

       <Button
           android:id="@+id/btInsertar"
           android:layout_width="fill_parent"
           android:layout_height="wrap_content"
           android:text="Insertar"
           android:layout_weight="5"/>

         <Button
           android:id="@+id/btVer"
           android:layout_width="fill_parent"
           android:layout_height="wrap_content"
           android:text="Ver"
           android:layout_weight="5"/>
    </LinearLayout>
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:weightSum="10">

        <EditText
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:id="@+id/etBuscar"
            android:layout_weight="4"
            android:hint="Ingrese el ID"
            android:numeric="integer"/>
        <Button
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="6"
            android:id="@+id/btBuscar"
            android:text="Buscar"/>


    </LinearLayout>

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:weightSum="10">

        <Button
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="5"
            android:id="@+id/btguardar"
```

```
            android:text="Guardar cambios"/>

        <Button
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="5"
            android:id="@+id/btEliminar"
            android:text="Eliminar"/>

        </LinearLayout>

</LinearLayout>
```

C.7 VerSQL (Activity)

```
package com.domotica;

import android.os.Bundle;
import android.app.Activity;
import android.app.Dialog;
import android.widget.TextView;

public class VerSQL extends Activity {
      TextView texto;
      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_ver_sql);
            texto=(TextView)findViewById(R.id.tvTextoVerSQL);

            SQLiteSMS informacion=new SQLiteSMS(this);

            try {
                  informacion.abrir();
            } catch (Exception e) {
                  Dialog d=new Dialog(this);
                  d.setTitle("Error al leer la base de datos");
                  d.show();
            }
       String info=informacion.mostrarDatos();
       informacion.cerrar();
       texto.setText(info);
      }

}
```

C.7.1 Interfaz usuario activity_ver_sql.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:id="@+id/tvTextoVerSQL"
        android:textSize="14sp"
        android:textColor="#FFFFFF"
      />


</LinearLayout>
```

C.8 Archivo AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.domotica"
    android:installLocation="preferExternal"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="10" />

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.SEND_SMS" />

    <application
        android:allowBackup="true"
        android:debuggable="true"
        android:icon="@drawable/icono_domotica"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.domotica.Domotica"
            android:label="@string/app_name"
            android:theme="@android:style/Theme.Wallpaper">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name="com.domotica.Vivienda"
            android:label="@string/title_activity_vivienda" >
        </activity>
        <activity
            android:name="com.domotica.VideoVigilancia"
            android:label="@string/title_activity_video_vigilancia">
        </activity>
        <activity
            android:name="com.domotica.Seguridad"
            android:label="@string/title_activity_seguridad">
        </activity>

        <activity
            android:name="com.domotica.Configuraciones"
            android:label="@string/title_activity_configuraciones" >
        </activity>
        <activity
            android:name="com.domotica.VerSQL"
```

```
            android:label="@string/title_activity_ver_sql"
            android:theme="@android:style/Theme.Dialog" >
        </activity>

    </application>

</manifest>
```

Anexo D

PROGRAMAS DE LAS PLACAS ARDUINO

D.1 Programa realizado para la placa Arduino Uno

A continuación se indica el código del programa de la placa Arduino Uno realizado para comunicarse con la aplicación de visión artificial.

```
#include <Servo.h>
int luces=8;
int ventilador=7;
Servo puerta;
Servo ventana;
char letra;

int angI=90;
int angF=0;
void setup(){
Serial.begin(9600);
pinMode(luces,OUTPUT);
pinMode(ventilador,OUTPUT);
puerta.attach(6);
ventana.attach(5);
puerta.write(angI);
ventana.write(angI);
delay(20);
}

void loop(){
if(Serial.available()>0){
letra=Serial.read();
if(letra=='a'){
Serial.println("Luces Encendidas");
digitalWrite(luces,HIGH);
}else if(letra=='b'){
Serial.println("Luces Apagadas");
digitalWrite(luces,LOW);
}else if(letra=='c'){
Serial.println("Ventilador Encendido");
digitalWrite(ventilador,HIGH);
}else if(letra=='d'){
Serial.println("Ventilador Apagado");
digitalWrite(ventilador,LOW);
```

```
}else if(letra=='e'){
Serial.println("Puerta Abierta");
puerta.write(angF);
delay(20);
}else if(letra=='f'){
Serial.println("Puerta Cerrada");
puerta.write(angI);
delay(20);
}else if(letra=='g'){
Serial.println("Ventana Abierta");
ventana.write(angF);
delay(20);
}else if(letra=='h'){
Serial.println("Ventana Cerrada");
ventana.write(angI);
delay(20);
}else{
Serial.println("Comando no Valido");
}
}
}
```

D.2 Programa realizado para la placa Arduino Nano

A continuación se indica el código del programa de la placa Arduino Nano realizado para comunicarse con la aplicación en Android.

```
#include <Servo.h>

//servidor Web 4 pines

#include "etherShield.h"
#include "ETHER_28J60.h"


int luz = 8;
int vent = 7;
int angC=90;//90 grados posicion cerrada
int angA=0;//0 grados posicion abierta

static byte mac[6] = {0x54, 0x55, 0x58, 0x10, 0x00, 0x24};

static byte ip[4] = {192, 168, 0, 2};

static int port = 80;

ETHER_28J60 e;
Servo servo1;
Servo servo2;
void setup()
```

```
{
  e.setup(mac, ip, port);
  pinMode(luz, OUTPUT);
  pinMode(vent, OUTPUT);
  servo1.attach(6);
  servo2.attach(5);
  servo1.write(angC);
  servo2.write(angC);
  delay(20);
}

void loop()
{
  char* params;
  if (params = e.serviceRequest())
  {
    e.print("<body bgcolor=\"#FFFF00\" vlink=\"#0000FF\" >");
    e.print("<H1>Control Domotico</H1>");
    e.print("<font size=5 >");
    e.print("Luces: <br>");

    if (strcmp(params, "?0001") == 0)//Compara strings de los links
    {
      digitalWrite(luz,HIGH);
      digitalWrite(vent,LOW);
      servo1.write(angC);
      servo2.write(angC);
      delay(20);
      e.print("<A HREF='?0000'>Desactivar</A>");
      e.print("<br><hr>Ventilador:<br> ");
      e.print("<A HREF='?0011'>Activar</A>");
      e.print("<br><hr>Puerta:<br> ");
      e.print("<A HREF='?0101'>Activar</A>");
      e.print("<br><hr>Ventana:<br> ");
      e.print("<A HREF='?1001'>Activar</A>");
      e.print("<br><hr>");
      e.print("</font>");
      e.print("</body>");
    }
    else if(strcmp(params, "?0010") == 0){
      digitalWrite(luz, LOW);
      digitalWrite(vent,HIGH);
      servo1.write(angC);
      servo2.write(angC);
      delay(20);
      e.print("<A HREF='?0011'>Activar</A>");
      e.print("<br><hr>Ventilador:<br> ");
      e.print("<A HREF='?0000'>Desactivar</A>");
      e.print("<br><hr>Puerta:<br> ");
      e.print("<A HREF='?0110'>Activar</A>");
```

```
  e.print("<br><hr>Ventana:<br> ");
  e.print("<A HREF='?1010'>Activar</A>");
  e.print("<br><hr>");
  e.print("</font>");
  e.print("</body>");

 }
  else if(strcmp(params, "?0011") == 0){
  digitalWrite(luz,HIGH);
  digitalWrite(vent,HIGH);
  servo1.write(angC);
  servo2.write(angC);
  delay(20);
  e.print("<A HREF='?0010'>Desactivar</A>");
  e.print("<br><hr>Ventilador:<br> ");
  e.print("<A HREF='?0001'>Desactivar</A>");
  e.print("<br><hr>Puerta:<br> ");
  e.print("<A HREF='?0111'>Activar</A>");
  e.print("<br><hr>Ventana:<br> ");
  e.print("<A HREF='?1011'>Activar</A>");
  e.print("<br><hr>");
  e.print("</font>");
  e.print("</body>");

 }else if(strcmp(params, "?0100") == 0){
  digitalWrite(luz,LOW);
  digitalWrite(vent,LOW);
  servo1.write(angA);
  servo2.write(angC);
  delay(20);
  e.print("<A HREF='?0101'>Activar</A>");
  e.print("<br><hr>Ventilador:<br> ");
  e.print("<A HREF='?0110'>Activar</A>");
  e.print("<br><hr>Puerta:<br> ");
  e.print("<A HREF='?0000'>Desactivar</A>");
  e.print("<br><hr>Ventana:<br> ");
  e.print("<A HREF='?1100'>Activar</A>");
  e.print("<br><hr>");
  e.print("</font>");
  e.print("</body>");

 }
  else if(strcmp(params, "?0101") == 0){////
    digitalWrite(luz,HIGH);
    digitalWrite(vent,LOW);
    servo1.write(angA);
    servo2.write(angC);
    delay(20);
  e.print("<A HREF='?0100'>Desactivar</A>");
  e.print("<br><hr>Ventilador:<br> ");
```

```
 e.print("<A HREF='?0111'>Activar</A>");
  e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?0001'>Desactivar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?1101'>Activar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
}
 else if(strcmp(params, "?0110") == 0){///

 digitalWrite(luz,LOW);
 digitalWrite(vent,HIGH);
 servo1.write(angA);
 servo2.write(angC);
 delay(20);
 e.print("<A HREF='?0111'>Activar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?0100'>Desactivar</A>");
 e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?0010'>Desactivar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?1110'>Activar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
}

 else if(strcmp(params, "?0111") == 0){//
  digitalWrite(luz,HIGH);
  digitalWrite(vent,HIGH);
  servo1.write(angA);
  servo2.write(angC);
  delay(20);
 e.print("<A HREF='?0110'>Desactivar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?0101'>Desactivar</A>");
 e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?0011'>Desactivar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?1111'>Activar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
}
 else if(strcmp(params, "?1000") == 0){

 digitalWrite(luz,LOW);
 digitalWrite(vent,LOW);
 servo1.write(angC);
```

```
 servo2.write(angA);
 delay(20);
 e.print("<A HREF='?1001'>Activar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?1010'>Activar</A>");
 e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?1100'>Activar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?0000'>Desactivar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
}
 else if(strcmp(params, "?1001") == 0){
  digitalWrite(luz,HIGH);
  digitalWrite(vent,LOW);
  servo1.write(angC);
  servo2.write(angA);
  delay(20);
 e.print("<A HREF='?1000'>Desactivar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?1011'>Activar</A>");
 e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?1101'>Activar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?0001'>Desactivar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
}
 else if(strcmp(params, "?1010") == 0){
  digitalWrite(luz,LOW);
  digitalWrite(vent,HIGH);
  servo1.write(angC);
  servo2.write(angA);
  delay(20);
 e.print("<A HREF='?1011'>Activar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?1000'>Desactivar</A>");
 e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?1110'>Activar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?0010'>Desactivar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
}
 else if(strcmp(params, "?1011") == 0){
  digitalWrite(luz,HIGH);
  digitalWrite(vent,HIGH);
```

```
  servo1.write(angC);
  servo2.write(angA);
  delay(20);
 e.print("<A HREF='?1010'>Desactivar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?1001'>Desactivar</A>");
  e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?1111'>Activar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?0011'>Desactivar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
}

 else if(strcmp(params, "?1100") == 0){
  digitalWrite(luz,LOW);
  digitalWrite(vent,LOW);
  servo1.write(angA);
  servo2.write(angA);
  delay(20);
 e.print("<A HREF='?1101'>Activar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?1110'>Activar</A>");
 e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?1000'>Desactivar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?0100'>Desactivar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
}


else if(strcmp(params, "?1101") == 0){
  digitalWrite(luz,HIGH);
  digitalWrite(vent,LOW);
  servo1.write(angA);
  servo2.write(angA);
  delay(20);
 e.print("<A HREF='?1100'>Desactivar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?1111'>Activar</A>");
 e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?1001'>Desactivar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?0101'>Desactivar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
```

```
}
else if(strcmp(params, "?1110") == 0){
  digitalWrite(luz,LOW);
  digitalWrite(vent,HIGH);
  servo1.write(angA);
  servo2.write(angA);
  delay(20);
 e.print("<A HREF='?1111'>Activar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?1100'>Desactivar</A>");
 e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?1010'>Desactivar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?0110'>Desactivar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
}
 else if(strcmp(params, "?1111") == 0){
  digitalWrite(luz,HIGH);
  digitalWrite(vent,HIGH);
  servo1.write(angA);
  servo2.write(angA);
  delay(20);
 e.print("<A HREF='?1110'>Desactivar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?1101'>Desactivar</A>");
 e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?1011'>Desactivar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?0111'>Desactivar</A>");
 e.print("<br><hr>");
 e.print("</font>");
 e.print("</body>");
}
 else {
  digitalWrite(luz,LOW);
  digitalWrite(vent,LOW);
  servo1.write(angC);
  servo2.write(angC);
  delay(20);
 e.print("<A HREF='?0001'>Activar</A>");
 e.print("<br><hr>Ventilador:<br> ");
 e.print("<A HREF='?0010'>Activar</A>");
 e.print("<br><hr>Puerta:<br> ");
 e.print("<A HREF='?0100'>Activar</A>");
 e.print("<br><hr>Ventana:<br> ");
 e.print("<A HREF='?1000'>Activar</A>");
 e.print("<br><hr>");
 e.print("</font>");
```

```
      e.print("</body>");

    }
    e.respond();
  }
}
```

Anexo E

CONSTRUCCIÓN DE LAS PLACAS DE POTENCIA

A continuación se indica el proceso de construcción de las placas de potencia. El software empleado para su diseño es Eagle 6.4.

E.1 Esquemático y Ruteado de las placas

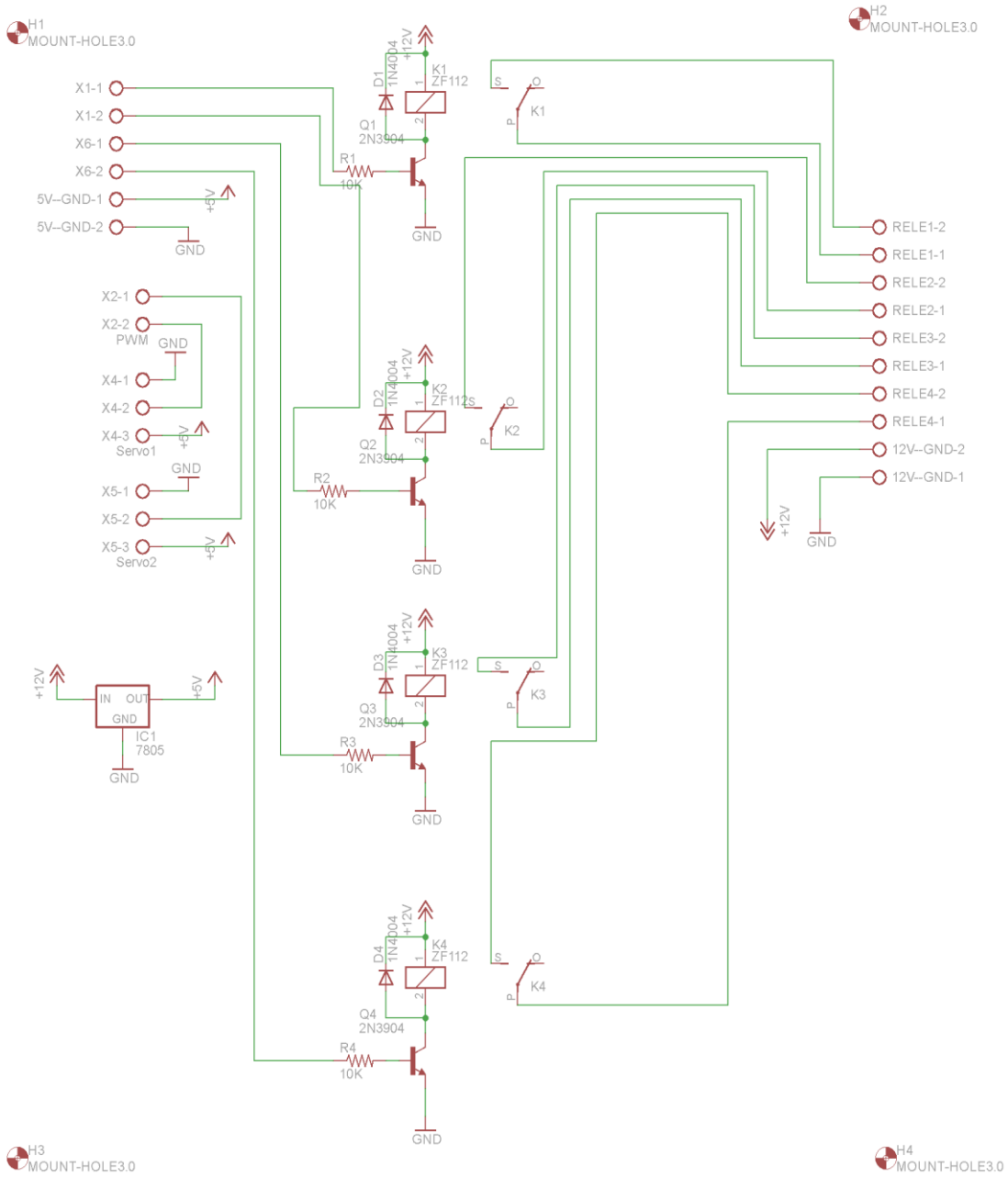


Figura 116. Esquemático de la placa 1.

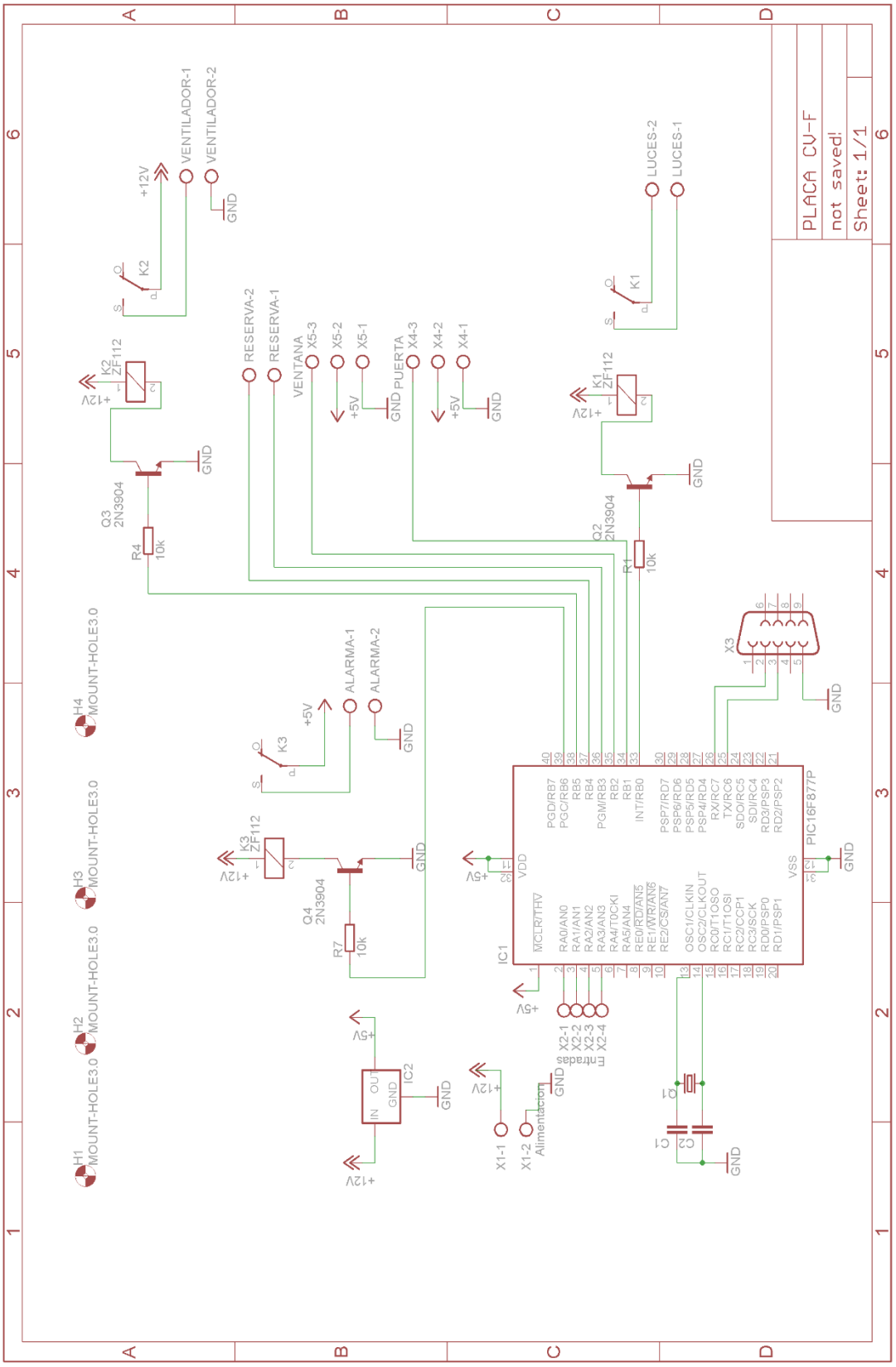


Figura 117. Esquemático de la placa 2.

E.2 Ruteado de las placas

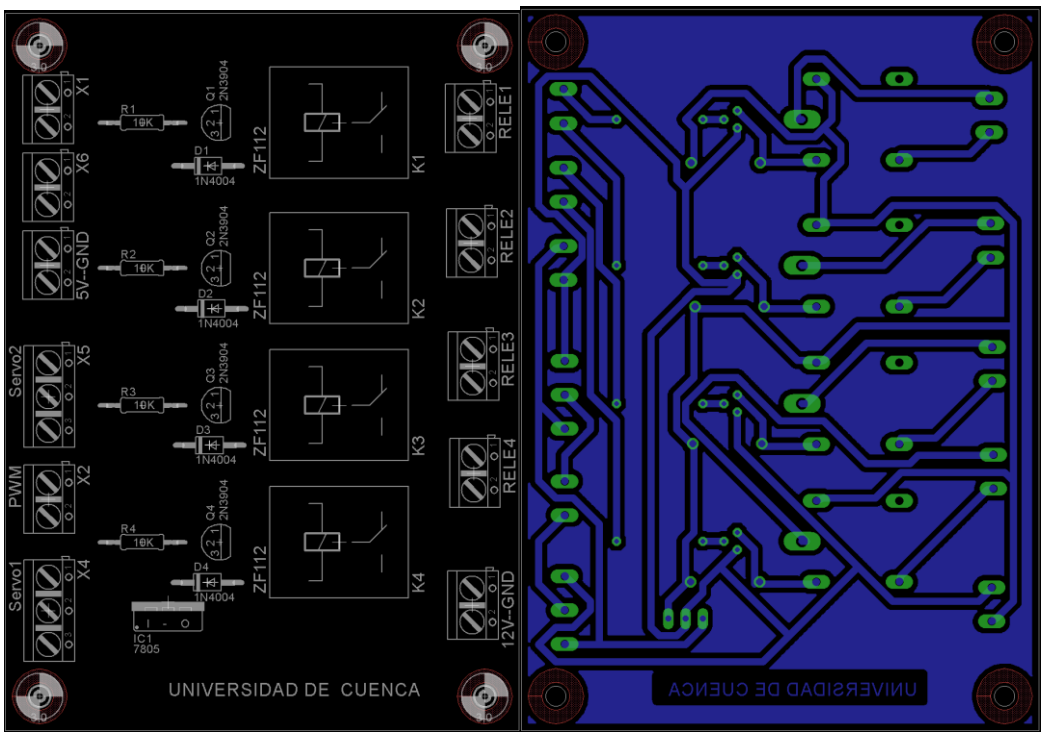


Figura 118. Ruteado de pistas de la placa 1.

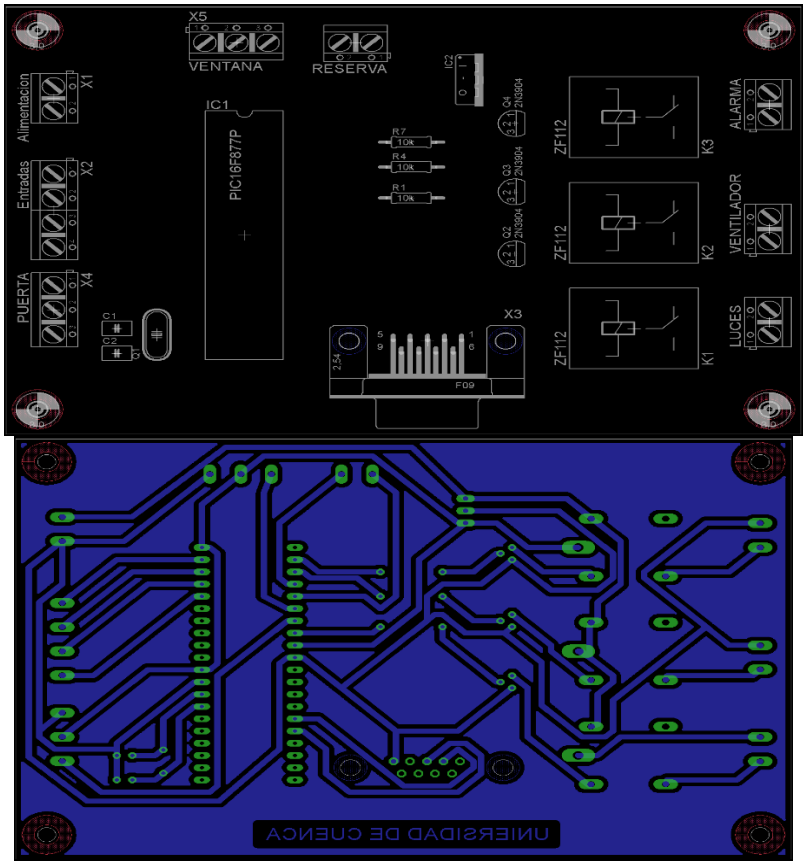


Figura 119. Ruteado de pistas de la placa 2.

A continuación se indica el proceso de construcción de las placas:

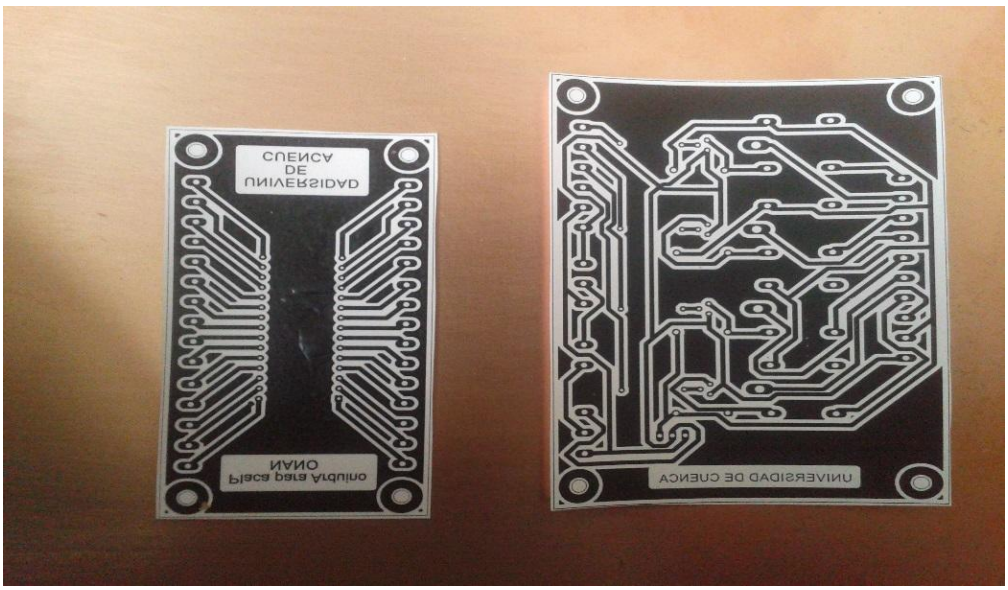


Figura 120. Circuito impreso antes de ser transferido a la baquelita.

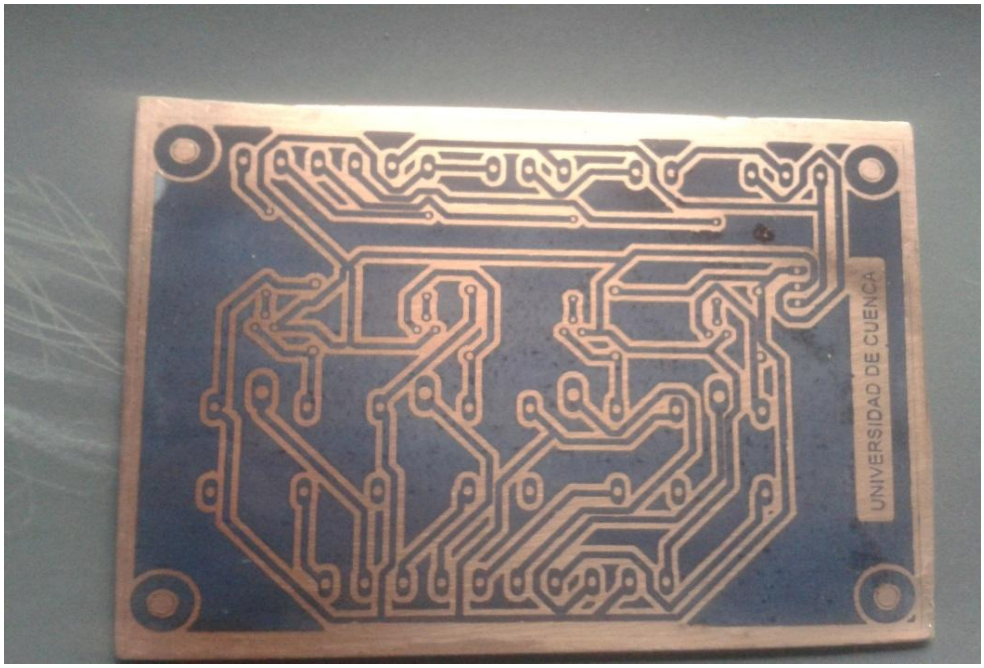


Figura 121. Circuito impreso en la baquelita.

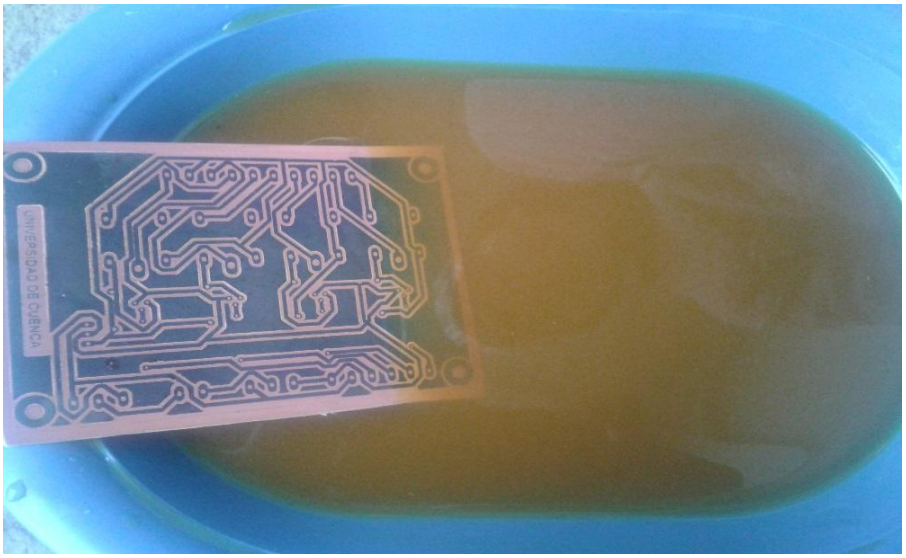


Figura 122. Ataque químico al circuito.

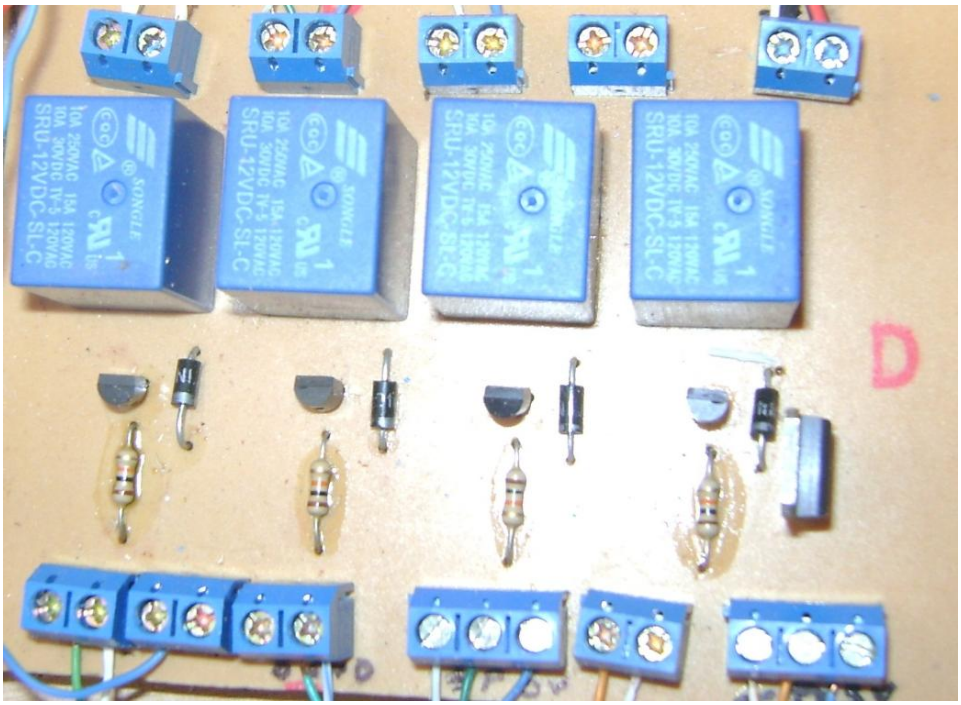


Figura 123. Placa 1 terminada con todos sus elementos montados.

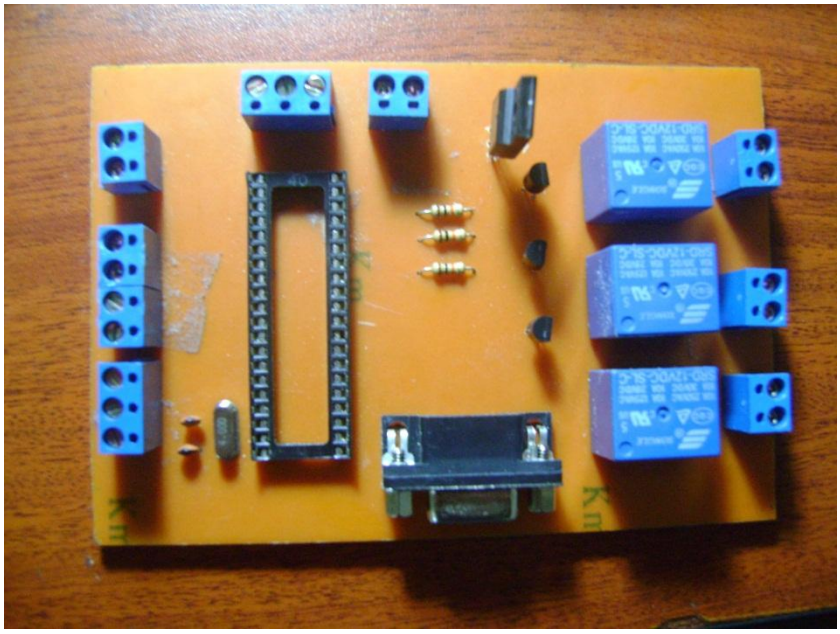


Figura 124. Placa 2 terminada con todos sus elementos montados.