TRABAJO ESPECIAL DE GRADO

LAZARILLO ELECTRÓNICO: DISEÑO DE UN SISTEMA DE DETECCIÓN DE OBSTÁCULOS

Presentado ante la ilustre
Universidad Central de Venezuela
por el Br. Moises Alejandro López Ribot
para optar al título
de Ingeniero Electricista

Caracas, Junio del 2017

TRABAJO ESPECIAL DE GRADO

LAZARILLO ELECTRÓNICO DISEÑO DE UN SISTEMA DE DETECCIÓN DE OBSTÁCULOS

TUTOR ACADÉMICO: Ing. Pedro Pinto

Presentado ante la ilustre
Universidad Central de Venezuela
por el Br. Br. Moises Alejandro López Ribot
para optar al título
de Ingeniero Electricista

Caracas, Junio 2017

CONSTANCIA DE APROBACIÓN

Caracas, 13 de junio de 2017

Los abajo firmantes, miembros del Jurado designado por el Consejo de Escuela de Ingeniería Eléctrica, para evaluar el Trabajo Especial de Grado presentado por el Bachiller Moises A. López R., titulado:

"LAZARILLO ELECTRÓNICO: DISEÑO DE UN SISTEMA DE DETECCIÓN DE OBSTÁCULOS"

Consideran que el mismo cumple con los requisitos exigidos por el plan de estudios conducente al Título de Ingeniero Electricista en la opción de Electrónica y Control, y sin que ello signifique que se hacen solidarios con las ideas expuestas por el autor, lo declaran APROBADO.

Prof. Panayotis Tremante
Jurado

Prof. Carlos Moreno
Jurado

Prof. Pedro Pinto
Tutor Académico

... A Dios y mi familia...

RECONOCIMIENTOS Y AGRADECIMIENTOS

A Dios y a la vida, por guiar mis pasos y permitirme aprender las cosas necesarias para seguir nuevos caminos.

A mi Padre Juan Pablo, Madre Maria Fernanda y hermano JP, que de no haber sido por ellos, este sueño nunca sería posible.

A los profesores de la escuela de Ingeniería Eléctrica por estar ahí a pesar de las dificultades que han tenido en estos últimos años y los más sabios en compartir todo su conocimiento posible.

Agradezco con mucho aprecio a mi tutor y profesor guía Pedro Pinto, por el tiempo dedicado durante el desarrollo del trabajo especial de grado y todas las dudas que aclaradas a lo largo del desarrollo del Trabajo Especial de Grado.

A los compañeros de infancia que siempre brindaron apoyo y a los compañeros universitarios que nos brindamos apoyo unos a otros para lograr nuestra meta en común.

A mi compañero de tesis y amigo Alejandro Gorrin, quien tuvo la paciencia y serenidad de afrontar junto a mi los problemas presentados en el transcurso del proyecto, a mi compañero y amigo Carlos González, quien me prestó gran parte de los componentes utilizados para el prototipo.

Y por último a mi Bella y muy especial Novia, Irene.

López R. Moises A.

LAZARILLO ELECTRÓNICO: DISEÑO DE UN SISTEMA DE DETECCIÓN DE OBSTÁCULOS

Tutor Académico: Prof. Pedro Pinto. Tesis. Caracas. U.C.V. Facultad de Ingeniería. Escuela de Ingeniería Eléctrica. Ingeniero Electricista. Opción: Electrónica. Año 2017.

Palabras Clave: Android Studio; OpenCV; Java; Python; Bluetooth; BeagleBone Black.

Resumen.- En el presente trabajo especial de grado, se busca diseñar un prototipo no comercial de Lazarillo Electrónico con la capacidad de detectar obstáculos presentes para las personas discapacitadas en una trayectoria cuyo destino es de su selección. Esta selección de destinos se realizó mediante la identificación de figuras de colores en un área delimitada, cada uno de los destinos dispondrá de una figura cuyo color identifica el destino en específico. Todo este proceso de detección de obstáculos se realizó mediante una cámara fotográfica que percibe el entorno constantemente. La implementación del prototipo se hizo en función de la disponibilidad de componentes y elementos, teniendo en cuenta un módulo controlador, un módulo Bluetooth, un puente H doble, dos motores, tres ruedas, dos baterías, un DC/DC, cables y una cámara fotográfica. La biblioteca utilizada para el procesamiento de imágenes, OpenCV, y las librerías para el control de las distintas funcionalidades a nivel de Hardware utilizadas en la placa de desarrollo, PWM, UART y GPIO, fueron programadas en lenguaje Python debido a su amplia información disponible. Finalmente se realizaron las distintas pruebas de detección de obstáculos para el Lazarillo considerando únicamente que la persona desee dirigirse hacia un objetivo en específico.

ÍNDICE GENERAL

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ACRÓNIMOS

JDK	Java Development Kit
UUID	Identificador Numero Universal
USB	Universal Serial Bus
PCB	Printed Circuit Board (Circuito Impreso)
BGR	Blue, Green & Red (Azul, Verde y Rojo)
HSV	Hue, Saturation & Value (Intensidad, Saturación y Valor)
FPS	Frames Per Second
TI	Texas Instrument
PWM	Pulse Width Modulation
GPIO	General Purpose Input/Output
UART	Universal Asynchronous Receiver Transmitter
eMMC	Embedded Micro Memory Card
API	Application Programming Interface
DBMS	DataBase Management System (Sistema manejador de base de datos)
APK	Android Application Package (Paquete de aplicación Android)
NDK	Native Development Kit (Kit de desarrollo nativo)
RFCOMM	Radio Frecuency Comunication
L2CAP	Logical Link Control and Adaptation Protocol

INTRODUCCIÓN

El prototipo de lazarillo electrónico a diseñar, para efectos de este Trabajo Especial de Grado, consta de principalmente dos subsistemas: El sistema de procesamiento de imágenes y el sistema de movimiento, existe un tercer sistema llamado el sistema de comunicación el cual el Br. Gorrin Alejandro realizó como tarea en su Trabajo Especial de Grado. El sistema de procesamiento de imágenes y el de movimiento tienen en común el BeagleBone Black, ya que en el mismo es donde se procesarán las imágenes y se controlaron el sentido de giro de las ruedas.

El procesamiento de imágenes se realiza por medio de la Biblioteca OpenCV versión 3.2.0, el control sobre las ruedas se realiza a través de un controlador o puente H doble gracias a las funcionalidades a nivel de Hardware de PWM y GPIO disponibles en la Biblioteca de Adafruit para el BeagleBone Black. El lenguaje de programación a utilizar en cada una de estas Bibliotecas y librerías fue Python, este posee una suficiente cantidad de información sobre los temas respectivos.

Debido a la gran cantidad de áreas que conforman este proyecto que son de suma importancia, se presenta en este Trabajo Especial de Grado todas las herramientas necesarias para el correcto entendimiento del prototipo de lazarillo electrónico no comercial, en el cual se verá incluido el avance realizado por el Br. Gorrin Alejandro sobre comunicación entre la persona discapacitada y el vehículo, representando datos relevantes junto a la implementación del lazarillo electrónico para su completo funcionamiento.

En el **Capítulo I** se expone el planteamiento del problema, junto con el objetivo general y los objetivos específicos del presente trabajo especial de grado.

En el **Capítulo II** se muestra el marco teórico, en donde se explican los conceptos, herramientas y métodos usados a lo largo del presente proyecto, tomando en cuenta los caminos tomados para resolver el problema y las variantes presentadas a lo largo de su ejecución.

En el **Capítulo III** se presenta la descripción general del proyecto, se describe la rutina de reconocimiento, la rutina de movimiento y la descripción del hardware necesaria para la implementación del lazarillo electrónico. De igual forma se presenta la descripción de la aplicación móvil, la configuración del módulo bluetooth y la descripción del sistema, los cuales son objetivos a tratar en el Trabajo Especial de Grado del Br Gorrin Alejandro.

En el **Capítulo IV** se muestran las pruebas realizadas para verificar la validación en la ejecución de la rutina, como también sus respectivos resultados.

Finalmente, se darán las conclusiones basadas en el resultado final y las posibilidades que ofrece el desarrollo de este proyecto para próximos proyectos a realizar.

CAPÍTULO I

DESCRIPCIÓN DEL PROYECTO

1.1. PLANTEAMIENTO DEL PROBLEMA

De acuerdo al último Censo realizado por el Instituto Nacional de Estadística (2011), en Venezuela se registró que la población en el territorio nacional supera los 27 millones de habitantes y, en base a estos resultados, el Instituto Nacional de Estadística (INE) se encargó de realizar un estudio de la cantidad de personas que en dicho censo poblacional presentan alguna deficiencia, condición o discapacidad.

De esta manera, se logró saber que de la población total existe una proporción importante del 5,38% (1.454.845) que afirma tener al menos una discapacidad, del cual un 1,7% (459.709) de esta población posee ceguera o alguna otra discapacidad visual.

Este porcentaje de invidentes se encuentra la capacidad de utilizar perros guías o lazarillos como acompañantes al momento en que la persona desee desplazarse, conduciendolos en vías públicas o auxiliándolas en el hogar.

Cualquier perro guía entrenado individualmente para proporcionar asistencia a una persona con una discapacidad. Los animales de servicio realizan algunas de las funciones y tareas que el individuo discapacitado no puede realizar por sí mismo. Los perros guía son un tipo de animal de servicio, usualmente utilizados por personas con limitaciones visuales. Estos perros están entrenados para guiar a personas ciegas en donde la elección del perro adecuado y el entrenamiento son las claves para un resultado óptimo.

En el caso de un lazarillo electrónico, el diseño de la comunicación que existe mediante éste y la persona discapacitada es de gran importancia ya que esta debe dar seguridad a la persona y evitar una dependencia con el equipo, limitando su uso para un proceso gradual de adaptación a la discapacidad, de esta forma, un lazarillo electrónico estaría presente en el proceso o avance de la deficiencia visual. Colocando la labor del lazarillo electrónico como una fase de aprendizaje y adaptación para la persona.

1.2 OBJETIVOS

1.2.1 Objetivo General

Diseñar un lazarillo electrónico con un sistema de detección de obstáculos en un área delimitada.

1.2.2 Objetivos Específicos

1. Identificar los principales objetos que representan obstáculos para las personas invidentes en sus áreas y entornos comunes.
2. Establecer la estructura y el Hardware para el lazarillo electrónico.
3. Determinar componentes, piezas y partes que integran los sistemas.
4. Implementar el lazarillo electrónico.
5. Desarrollar el sistema de deteccion de obstaculos para un área delimitada.
6. Realizar pruebas y ensayos del sistema.

CAPÍTULO II

MARCO TEÓRICO

2.1 Ceguera

Es la ausencia total de visión, no hay percepción de luz. La pérdida de la visión puede suceder de manera repentina o con el paso del tiempo, derivando sus especificaciones en los tipos de ceguera, que se produce debido a una afectación congénita o adquirida.

De acuerdo a la Organización Mundial de la Salud, la función visual se subdivide en cuatro niveles:

1. visión normal;
2. discapacidad visual moderada;
3. discapacidad visual grave;
4. ceguera.

El segundo y tercer nivel alude a aquellas personas que con la mejor corrección posible podrían ver o distinguir, aunque con gran dificultad, algunos objetos a una distancia muy corta. En la mejor de las condiciones, algunas de ellas pueden leer la letra impresa cuando ésta es de suficiente tamaño y claridad, pero, generalmente, de forma más lenta, con un considerable esfuerzo y utilizando ayudas especiales.

El cuarto nivel se refiere a cuando la persona no ve ni siente absolutamente nada, ni siquiera luz ni su reflejo (resplandor). Más específicamente, una persona con ceguera es aquella que no ve nada en absoluto o solamente tiene una ligera percepción de luz (puede ser capaz de distinguir entre luz y oscuridad, pero no la forma de los objetos).

2.2 Lazarillo

Lazarillo es el diminutivo de Lázaro. El término se utiliza para nombrar a la persona o al animal que guía, dirige o acompaña a quien necesita de ayuda para desplazarse. Este uso del concepto procede de la novela *"Lazarillo de Tormes"*, publicada en el siglo XVI.

En dicha publicación, de estilo epistolar, se narra en primera persona la historia de Lazarillo, quien actuaba como guía de un hombre ciego. A partir de entonces, con el paso de los años, la noción de lazarillo comenzó a asociarse a esta labor.

2.2.1. Perro Lazarillo

Es frecuente, aún en la actualidad, que los ciegos cuentan con el apoyo de un perro lazarillo a la hora de desplazarse. Estos perros de asistencia son entrenados en centros especializados para que puedan asistir a las personas no videntes, conduciendolas en la vía pública y auxiliándolas en el hogar.

Entre los perros que habitualmente son entrenados para que ejerzan de lazarillos de personas con problemas de visión se encuentran los de la raza Labrador Retriever, ya que tienen un carácter muy sosegado y calmado, fundamental para la labor que desempeñan.

El perro lazarillo o perro guía tiene la capacidad de advertir los peligros que ciertas barreras arquitectónicas pueden representar para su dueño. Por eso el no vidente lleva al perro con una correa y el animal es quien dirige la caminata, esquivando los obstáculos que podrían hacer que la persona tropiece o se golpee.

2.2.2. Comunicación con el Perro Lazarillo

2.2.2.1. El uso del arnés

El arnés proporciona un canal de comunicación entre el perro y la persona que lo maneja. El perro y la persona deberán adoptar una posición correcta para moverse de forma segura y eficiente como unidad. La posición normal es el perro situado a la izquierda de la persona con aproximadamente tres cuartas partes de su cuerpo por delante de esta, de esta manera se tiene un tiempo de reacción que permitirá a la persona que lo maneja darse cuenta de los movimientos del perro, pudiendo intuir lo que se encuentra delante de él y reaccionar de manera adecuada ante cualquier cambio de dirección del perro o parada repentina. Esto justifica el hecho de que el asa sea de metal y de distintas longitudes para permitir obtener la posición correcta.

Es necesaria una tensión entre el perro y la persona que lo maneja sin llegar a ser fuerte, dado que resultaría extremadamente incómoda y podría llegar a producir problemas/dolores de espalda e incluso deteriorar todos los aspectos de trabajo del perro guía (incremento de los niveles de estrés e incremento de errores, dificultad de control, entre otros). El grado de tensión varía de una persona a otra y se relaciona con el peso, agilidad, acción refleja o equilibrio.

Todo esto se debe evaluar al asignar un perro a una persona y comprobar la compatibilidad durante el entrenamiento de la unidad perro – guía.

2.2.2.2 Ordenes

- Órdenes de dirección: (derecha, atrás, izquierda, avanza…)

- Órdenes de control: (no, deja, calla, mal) se utilizan para controlar conductas específicas, por ejemplo, detenerse, esperar, distracciones, mala concentración.

Todas estas órdenes utilizadas con movimientos y acciones del cuerpo, pies y manos, son interpretadas con mayor facilidad y se refuerzan por el uso de incentivos.

Existen otras órdenes útiles para el ciego como puede ser la búsqueda de un sitio libre en el autobús, la búsqueda de una parada de metro/autobús. La aproximación a la puerta del autobús y la detención para que el ciego pueda preguntar por el número del recorrido.

También se les enseñan órdenes como sigue, útil por ejemplo en restaurantes cuando te llevan a una mesa o cuando se realiza un desplazamiento en grupo.

Dependiendo de las actividades propias del invidente se le enseña al conjunto del perro y la persona a realizar de manera adecuada recorridos habituales (ir a trabajar, visitar a la familia) en metro, autobús o andando. Ambos aprenderán a coger las líneas adecuadas y a orientarse una vez salen de nuevo a la vía pública.

El adiestrador del perro guía siempre está disponible para el usuario ante cualquier necesidad o duda respecto a su trabajo en común con el perro.

2.2.3. El principio de la línea recta.

Hay que tener claro desde un principio que es la persona quien maneja al perro la responsable de su orientación y que no se debe basar en la capacidad del perro para conseguir el objetivo deseado.

Una de las características de la movilidad con perro guía es que la persona ciega no tiene contacto físico con el entorno, se eliminan las referencias táctiles que puede obtener con el bastón (línea del edificio, farolas, etc.) y ha de utilizar las referencias auditivas o cambios de superficie para poder determinar su situación en el entorno. Los únicos puntos de referencia y orientación, que se

mantienen estables en el entorno, son el tráfico, los bordillos, los cambios de superficie, otros sonidos (tiendas, etc.) u olores (panadería, kiosco, etc.).

Por esta razón se adiestra al perro guía siguiendo el principio de la línea recta, para facilitar la movilidad independiente y segura de la persona ciega. El perro responderá a las órdenes de dirección de la persona que lo maneja y una vez tomada la dirección mantendrá una línea recta hasta nueva orden o hasta que tenga que desviarse o detenerse debido a influencias ambientales.

Cuando caminen por la acera se mantendrá siempre por el centro de la misma y cuando ésta no exista la línea de desplazamiento será cercana y paralela al borde de la calzada.

2.3 Imagen Digital

Una imagen digital es aquella que puede ser obtenida a través de una cámara fotosensible, el valor asignado al conjunto de pixeles se representan como una serie de unos y ceros. La calidad de la imagen puede variar principalmente de acuerdo a las siguientes características:

-Sensibilidad del pixel.
-Resolución soportada por la cámara.

Esta información es interpretada correctamente por una computadora, y una vez guardada, ésta puede ser procesada para la recolección de alguna información de interés disponible en la imagen. La transformación de una fotografía analógica en una digital es lo que se conoce como Digitalización, siendo éste siempre el primer paso para el procesamiento de imágenes.

El píxel es la partícula elemental de una imagen, cada uno de estos elementos contiene la información del color percibido, dicho color es una mezcla de los colores rojo, verde y azul, siendo estos últimos los componentes elementales de la

Luz. La resolución de una imagen digital es la concentración o densidad pixeles que tiene; Tener mayor resolución se traduce en obtener una imagen más detallada o con mejor calidad visual.

La representación de las fotografías digitales como archivo informático para su manipulación, consiste en dividir la imagen como una malla de filas y columnas de píxeles, en otras palabras es una función bidimensional de intensidad de iluminación $f(x,y)$, donde "x" y "y" representan las coordenadas espaciales de la imagen. El valor de $f(x,y)$ evaluada en el punto (x,y) es proporcional a la intensidad de iluminación de la fotografía en ese punto (x,y).

En la figura 1 se muestra un ejemplo de una matriz de píxeles para una imagen:

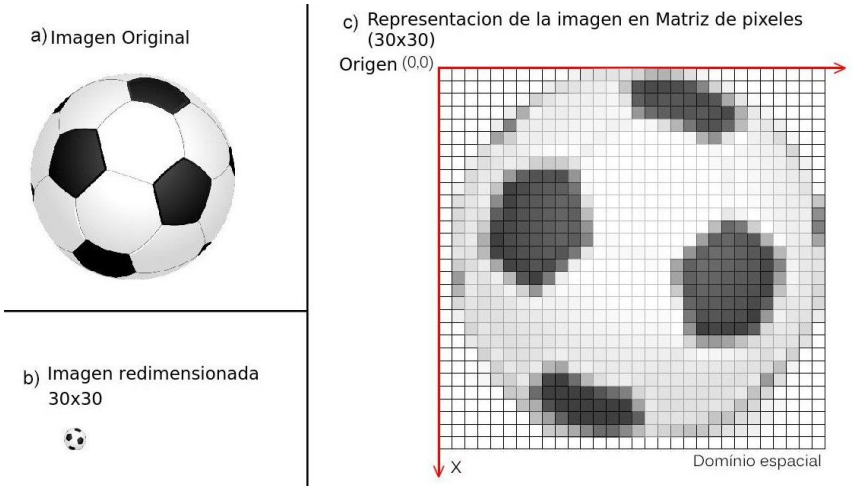

Figura 1: Matriz de píxeles 30x30, a)Imagen original, b) Imagen redimensionada 30x30, c) Representación de la imagen en Matriz de píxeles 30x30.

2.3.1. Procesamiento de Imágenes Digitales.

El principal objetivo del procesamiento de imagenes es la extraccion de informacion, haciendo uso de herramientas informáticas como un conjunto de técnicas aplicadas a la matriz de píxeles es posible recolectar la información de interés disponible en la imagen. También se realiza el procesamiento de imágenes para mejorar la calidad de su visualización.

2.3.1.1 Etapas.

Para la obtención de información a partir del mundo analógico, es necesario las escenas de interés a estudiar se hagan pasar a través de las siguientes etapas:

1. **Captura:** en esta parte se considera el diseño de las propiedades de la captura, como lo son: tipo de camara, resolucion, cantidad de luz en el ambiente, etc.

2. **Pre-Procesamiento:** en esta parte se busca reducir el entorno que no es de interés para el problema. Los principales objetivos del procesamiento previo de la imagen se realizan principalmente con la aplicación de filtros. Las principales funciones de estos filtros permiten:

- Suavizar la imagen: Básicamente consiste en reducir la cantidad de variaciones de intensidad entre píxeles vecinos sustituyendo el pixel central por el equivalente del método aplicado de la matriz de dichos píxeles vecinos. Esta funcionalidad posee una gran cantidad de métodos diferentes que varían principalmente de acuerdo al tipo filtrado aplicado o Kernel utilizado en la imagen. Entre los tipos de filtrado existen: Filtro Gaussiano, filtro promedio, filtro Paso bajo y filtro Mediana.

Un ejemplo de la imagen resultante en la aplicación de un Kernel Gaussiano se puede observar en la siguiente figura a continuación:

Figura 2: Suavizado con filtro Gaussiano. A la izquierda se muestra la imagen original, a la derecha la imagen filtrada.

- Eliminar ruido: Eliminar aquellos píxeles cuyo nivel de intensidad es muy diferente al de sus vecinos y cuyo origen puede estar tanto en el proceso de adquisición de la imagen como en el de transmisión. La información no deseada introducida en la imagen resultante obtenida desde el mundo analógico hasta su conversión a formato digital, se debe a una serie de factores que se encuentran en los distintos medios de transmisión. Para esto se utilizan filtros pasa bajo, manteniendo sólo los píxeles cuyos vecinos no varíen bruscamente sus valores respecto al pixel central.

Ejemplos de adquisición de ruido en una imagen:

1. Ruido introducido por las películas de la cámara como sensor de imágenes.
2. Ruido Térmico.
3. Ruído fotoeléctrico.
4. Ruido Impulsivo o sal y pimienta.
5. Ruido aditivo.
6. Ruido multiplicativo.
7. Ruido frecuencial.

Un ejemplo de ruido impulsional se puede observar en la siguiente imagen:

Figura 3: Ruido “Sal y Pimienta” en una imagen.

- Realzar bordes: Destacar los bordes que se localicen en la imagen.

-Detectar bordes: Detectar los pixeles donde se produce un cambio brusco en la función intensidad. El tipo de filtro utilizado para detectar bordes es de alta frecuencia, puesto que estas corresponden en las imágenes a los cambios bruscos de densidad. Los cambios bruscos de los valores de píxeles en una imagen se pueden notar fácilmente cuando existe contraste en la imagen. Este método de aplicación es una herramienta fundamental en el procesamiento de imágenes y en visión computarizada, particularmente en las áreas de detección y extracción de características, que tiene como objetivo la identificación de puntos en una imagen digital en la que el brillo de la imagen cambia drásticamente, o más formalmente, tiene discontinuidades. De los métodos existentes conocidos para realizar este procesamiento se conoce: Los métodos por gradiente y el método “Canny Edge”.

En la siguiente figura se puede observar un ejemplo de la aplicación de un método de detección de bordes:

Figura 4: Método Canny Edge Detection aplicado a una fotografía.

Los filtros se consideran como operaciones que se aplican a los pixeles de una imagen digital para optimizarla, enfatizar cierta información o conseguir un efecto especial en ella.

3. **Segmentación:** En esta etapa del proceso se busca simplificar y/o cambiar la representación de la imagen en otra más significativa y más fácil de analizar. La segmentación se usa tanto para localizar objetos como para encontrar los límites de estos dentro de una imagen, en otras palabras, es el proceso de asignación de una etiqueta a cada pixel de la imagen de forma que los píxeles que compartan la misma etiqueta también tendrán ciertas caracteristicas visuales similares. Esta etapa también involucra utilizar la menor cantidad de bits posibles para representar la imagen sin comprometer la calidad de la misma.

En la siguiente figura se muestra un ejemplo de una imagen segmentada,

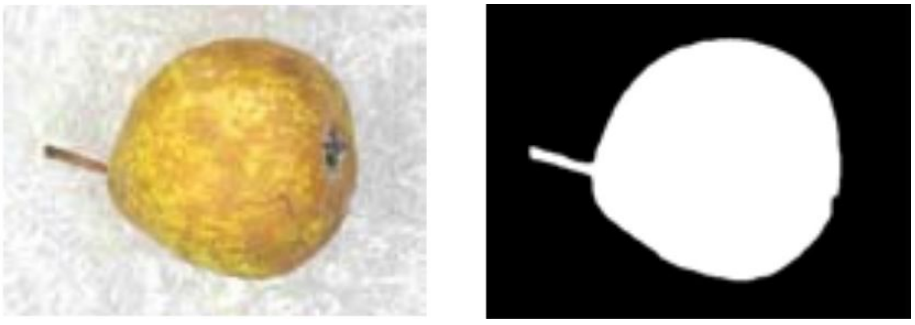

Figura 5: Segmentación de una imagen.

4. **Extracción de características:** La extracción de información de las fotografías a través del procesamiento digital constituye hoy en día un inmenso campo de estudio e investigación en diversas disciplinas con múltiples aplicaciones. Una imagen contiene una gran cantidad de datos la mayoría de los cuales proporciona muy poca información para interpretar la escena. Un sistema que incorpore Visión Computarizada debe, en primer paso, extraer de la forma más eficaz y robusta posible determinadas características que nos proporcionen la maxima informacion posible.

5. Identificación de objetos: Por último para el procesamiento de una imagen se utiliza un modelo de toma de decisión para deducir a qué categoría pertenece cada objeto, contorno o silueta identificada. Posterior a esta última etapa se pueden utilizar algoritmos genéticos de clasificación para deducir de una manera más eficiente y eficaz el tipo de objeto identificado.

2.3.2 Espacios de Color

Un espacio de color es un sistema de interpretación del color, es decir, una organización específica de los colores en una imagen o video. Depende del modelo de color en combinación con los dispositivos físicos que permiten las representaciones reproducibles de color, por ejemplo las que se aplican en señales analógicas (televisión a color) o representaciones digitales. Un espacio de color puede ser arbitrario, con colores particulares asignados según el sistema y estructurados matemáticamente [12].

Un modelo de color es un modelo matemático abstracto que describe la forma en la que los colores pueden representarse como tuplas de números, normalmente como tres o cuatro valores o *componentes de color*, por ejemplo RGB, CMYK y HSV son modelos de color. Sin embargo, un modelo de color que no tiene asociada una función de mapeo a un espacio de color absoluto es más o menos un sistema de color arbitrario sin conexión a un sistema de interpretación de color.

Añadiendo cierta función de mapeo entre el modelo de color y un espacio de color de referencia se obtiene una "huella" en el espacio de color de referencia. A esta "huella" se la conoce como gama de color y, en combinación con el modelo de color, define un nuevo espacio de color. Por ejemplo, Adobe RGB y sRGB son dos espacios de color absolutos diferentes basados en el modelo RGB.

En el sentido más genérico de la definición dada, los espacios de color se pueden definir sin el uso de un modelo de color. Estos espacios, como Pantone, son un conjunto de nombres o números definidos por la existencia de un conjunto correspondiente de muestras de color físico. Este artículo se centra en el concepto del modelo matemático.

2.3.2.1 Espacio de color RGB

RGB (sigla en inglés de *red, green, blue*, en español rojo, verde y azul) es la composición del color en términos de la intensidad de los colores primarios de la luz. Dicho espacio de color es un modelo de color basado en la síntesis aditiva, con el que es posible representar un color mediante la mezcla por adición de los tres colores de luz primarios. El modelo de color RGB no define por sí mismo lo que significa exactamente rojo, verde o azul, por lo que los mismos valores RGB pueden mostrar colores notablemente diferentes en distintos dispositivos que usen este modelo de color. Aunque utilicen un mismo modelo de color, sus espacios de color pueden variar considerablemente.

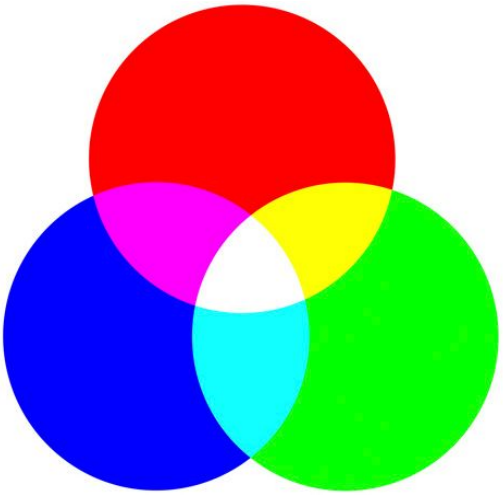

Figura 6: Colores primarios de la Luz.

2.3.2.2 Espacio de Color HSV

El modelo HSV (del inglés *Hue, Saturation, Value* – Matiz, Saturación, Valor), también llamado HSB (*Hue, Saturation, Brightness* – Matiz, Saturación, Brillo), define un modelo de color en términos de sus componentes.

Sus componentes se definen con las siguientes características:

-Matiz: Se representa como un grado de ángulo cuyos valores posibles van de 0 a 360° (aunque para algunas aplicaciones se normalizan del 0 al 100%). Cada valor corresponde a un color. Ejemplos: 0 es rojo, 60 es amarillo y 120 es verde. De forma intuitiva se puede realizar la siguiente transformación para conocer los valores básicos RGB:

Dicho componente dispone de 360 grados donde se dividen los 3 colores RGB, eso da un total de 120º por color, sabiendo esto podemos recordar que el 0 es rojo RGB(1, 0, 0), 120 es verde RGB(0, 1, 0) y 240 es azul RGB(0, 0, 1).

Visualmente se puede observar la siguiente tabla tabla:

Tabla 1: Componentes RGB equivalente en intensidad H.

Componentes R,G,B y Color	Ángulo de Intensidad
(0,0,0)-----Rojo	0 º
(1,1,0)----Amarillo	60º
(0,1,0)-----Verde	120º
(0,1,1)-----Cian	180º
(0,0,1)----Azul	240 º
(1,0,1)----Fucsia	300 º
(1,0,0)----Rojo	360 º =0 º

Para colores mixtos se utilizan los grados intermedios, el amarillo, RGB(1, 1, 0) está entre rojo y verde, por lo tanto 60º.

-Saturación: Se representa como la distancia al eje de brillo negro-blanco. Los valores posibles van del 0 al 100%. A este parámetro también se le suele llamar "pureza" por la analogía con la pureza de excitación y la pureza colorimétrica de la colorimetría. Cuanto menor sea la saturación de un color, mayor tonalidad grisácea habrá y más decolorado estará. Por eso es útil definir la *insaturación* como la inversa cualitativa de la saturación.

La barra mostrada en la figura 7 se muestra el color rojo con distintos valores de saturación:

Figura 7: Valores de saturación del color Rojo.

-Valor: Se denomina “Valor” a la amplitud de la Luz que define el color; más cerca del negro, más bajo es el valor, lo cual ocurre todo lo contrario para el color Blanco. Solo existen dos valores: Blanco y negro. Los grises, que son tonos del blanco y negro, no son valores.

Se suelen representar como una altura a lo largo de un eje que va desde el blanco hasta el negro. Los valores posibles van del 0 al 100%. 0 siempre es negro. Dependiendo de la saturación, 100 podría ser blanco o un color más o menos saturado.

En la figura 8 se puede observar el Cono del espacio de color HSV:

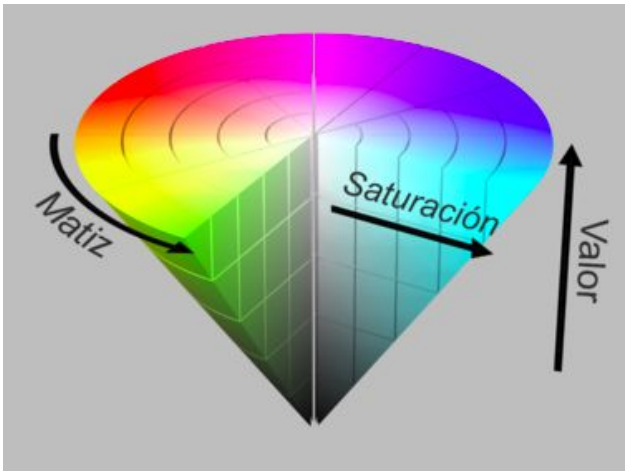

Figura 8: Cono del espacio de color HSV.

2.4. OpenCV

OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Desde que apareció su primera versión alfa en el mes de enero de 1999, se ha utilizado en infinidad de aplicaciones. Desde sistemas de seguridad con detección de movimiento, hasta aplicaciones de control de procesos donde se requiere reconocimiento de objetos. Esto se debe a que su publicación se da bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación con las condiciones en ella expresadas.

OpenCV es multiplataforma, tiene versiones para C++, Python, Java, entre otras. Posee versiones para GNU/Linux, Mac OS X y Windows, contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como

reconocimiento de objetos (reconocimiento facial), calibración de cámaras, visión estérea y visión robótica.

2.5. Android

Es una Sistema Operativo de software para dispositivos móviles que incluye aplicaciones de base.

Android es un conjunto de herramientas y aplicaciones vinculadas a una distribución Linux para dispositivos móviles. Por sí solo no es un Sistema Operativo, Android es de código abierto, gratuito y no requiere pago de licencias.

Android es una plataforma de código abierto para dispositivos móviles que está basada en Linux y desarrollada por Open Handset Alliance, los primeros teléfonos con Android aparecieron en el segundo semestre de 2008 y compañías poderosas como LG, Motorola y HTC diseñaron alguno de los prototipos que incorporaron el Sistema Android.

Es una stack de software para dispositivos móviles que incluye un Sistema Operativo, Middleware y aplicaciones de base. Los desarrolladores pueden crear aplicaciones para la plataforma usando el SDK de Android. Las solicitudes se han escrito utilizando el lenguaje de programación Java y se ejecutan en Dalvik, una máquina virtual personalizada que se ejecuta en la parte superior de un núcleo de Linux.

2.5.1. Arquitectura

- **Aplicaciones:** incluyen un cliente de correo electrónico, programa de SMS, calendario, mapas, navegador, contactos y otros. Todas ellas escritas en Java.

- **Marco de trabajo 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; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede luego hacer uso de esas capacidades.

- **Bibliotecas:** incluye un conjunto de bibliotecas de C/C++ usadas por varios componentes del sistema.

- **Runtime de Android:** incluye un set de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las bibliotecas base del lenguaje Java. Cada aplicación Android corre su propio proceso, con su propia instancia de la máquina virtual.

- **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. También actúa como capa de abstracción entre el hardware y el resto de la pila de software.

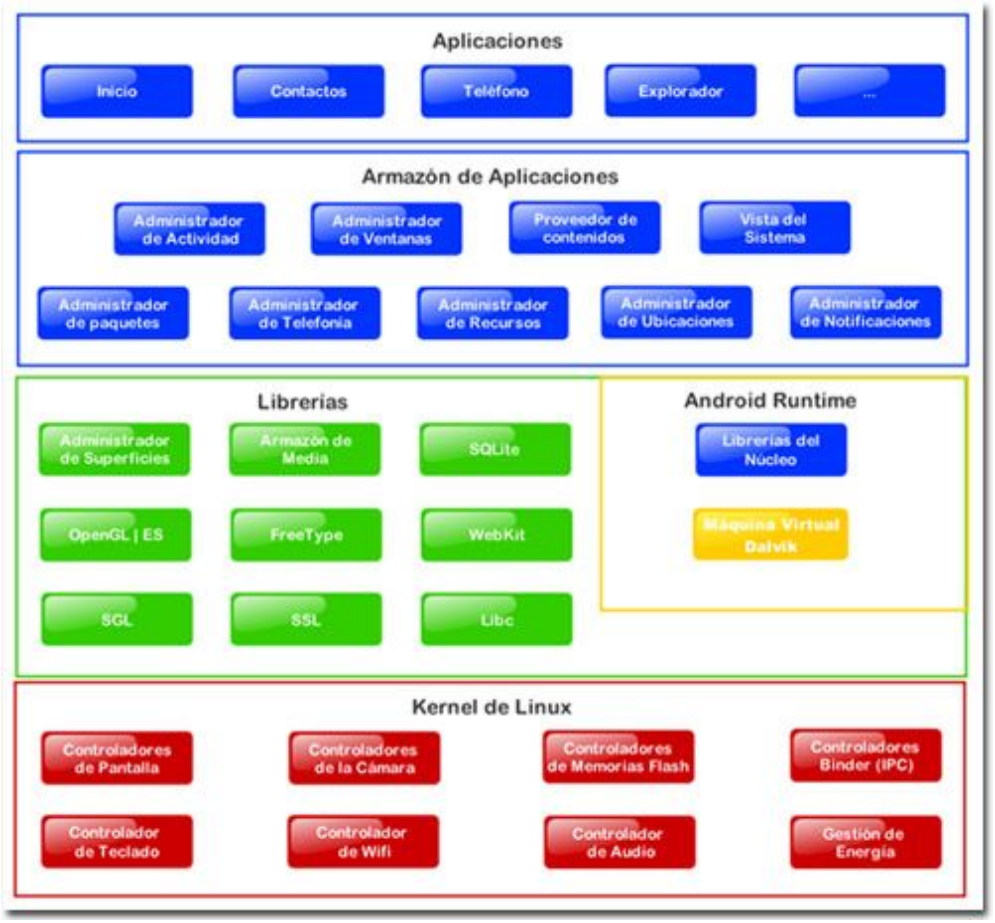

Figura 9: Arquitectura de Android

2.5.2 Evolución

Para el desarrollo en el entorno Android, se debe tener en cuenta las características del equipo, por esto se debe verificar la versión del equipo y los niveles de API de acuerdo a la aplicación a desarrollar.

Tabla 2: Evolución de Android

Nombre Código	Número de Versión	Fecha de lanzamiento	Nivel de API
Apple Pie	1.0	23 de septiembre 2008	1
Banana Bread	1.1	9 de febrero 2009	2
Cupcake	1.5	27 de abril de 2009	3

Donut	1.6	15 de septiembre de 2009	4
Eclair	2.0 - 2.1	26 de octubre de 2009	5-7
Froyo	2.2 - 2.2.3	20 de mayo 2010	8
Gingerbread	2.3 - 2.3.7	6 de diciembre 2010	9-10
Honeycomb	3.0 - 3.2.6	22 de febrero de 2011	11-13
Ice Cream Sandwich	4.0 - 4.0.4	18 de octubre 2011	14-15
Jelly Bean	4.1–4.3.1	9 de julio de 2012	16-18
KitKat	4.4 - 4.4.4	31 de octubre de 2013	19-20
Lollipop	5.0–5.1.1	12 de noviembre de 2014	21-22
Marshmallow	6.0–6.0.1	5 de octubre de 2015	23
Nougat	7.0 - 7.1.2	22 de agosto de 2016	24-25
Android 0	8.0		

2.5.3. Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJ IDEA. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece funciones que permiten aumentar la productividad durante la compilación de apps para Android, como las siguientes:

2.5.3.1. Características

- Sistema de compilación flexible basado en Gradle.
- Un emulador rápido con varias funciones.

- Un entorno unificado en el que puedes realizar desarrollos para todos los dispositivos Android.
- Instant Run, para aplicar cambios mientras tu app se ejecuta sin la necesidad de compilar un nuevo APK.
- Integración de plantillas de código y GitHub, para ayudarte a compilar funciones comunes de las apps e importar ejemplos de código.
- Gran cantidad de herramientas y frameworks de prueba.
- Herramientas Lint para detectar problemas de rendimiento, uso, compatibilidad de versión, etc.
- Compatibilidad con C++ y NDK.
- Soporte integrado para Google Cloud Platform, que facilita la integración de Google Cloud Messaging y App Engine.
- Integración de ProGuard y funciones de firma de aplicaciones.
- Renderizado en tiempo real.
- Consola de desarrollador: consejos de optimización, ayuda para la traducción, estadísticas de uso.
- Refactorización específica de Android y arreglos rápidos.
- Un editor de diseño enriquecido que permite a los usuarios arrastrar y soltar componentes de la interfaz de usuario.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versiones, y otros problemas.
- Plantillas para crear diseños comunes de Android y otros componentes.
- Soporte para programar aplicaciones para Android Wear.
- Soporte integrado para Google Cloud Platform, que permite la integración con Google Cloud Messaging y App Engine.
- Un dispositivo virtual de Android que se utiliza para ejecutar y probar aplicaciones.

2.5.3.2. Google APIs

Una API (siglas de 'Application Programming Interface') es un conjunto de reglas (código) y especificaciones que las aplicaciones pueden seguir para

comunicarse entre ellas: sirviendo de interfaz entre programas diferentes de la misma manera en que la interfaz de usuario facilita la interacción humano-software.

Las API pueden servir para comunicarse con el sistema operativo (WinAPI), con bases de datos (DBMS) o con protocolos de comunicaciones (Jabber/XMPP). En los últimos años, se han sumado múltiples redes sociales (Twitter, Facebook, Youtube, Flickr, LinkedIn, etc) y otras plataformas online (Google Maps, WordPress…), lo que ha convertido el social media marketing es más sencillo, más rastreable y, por tanto, más rentable.

Las API son valiosas, ante todo, porque permiten hacer uso de funciones ya existentes en otro software (o de la infraestructura ya existente en otras plataformas), reutilizando así código que se sabe que está probado y que funciona correctamente. En el caso de herramientas propietarias (es decir, que no sean de código abierto), son un modo de hacer saber a los programadores de otras aplicaciones cómo incorporar una funcionalidad concreta sin por ello tener que proporcionar información acerca de cómo se realiza internamente el proceso.

2.5.3.3. Java

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (conocido en inglés como *WORA*, o "*write once, run anywhere*"), lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser compilado para correr en otra. Java es, a partir de 2012, uno de los lenguajes de programación más populares en uso, particularmente para aplicaciones de cliente-servidor de web, con unos 10 millones de usuarios reportados.

Como todo lenguaje de programación, Java se utiliza para crear aplicaciones y procesos que funcionen en multitud de dispositivos. La versión estándar de Java es responsable de varias aplicaciones muy conocidas como jDownloader o Vuze.

Las aplicaciones Java se comunican con la máquina virtual Java, y no con el sistema operativo, lo cual permite a los programadores desentenderse de la compatibilidad con el hardware: esta es tarea para la máquina virtual de Java.

Los applets Java no son más que pequeñas aplicaciones que se ejecutan en navegador web, y requieren tener instalado el plugin Java correspondiente (se incluye en la instalación estándar de Java). Estos applets, incrustados en páginas web, permitían hacer cosas entonces imposibles para HTML, como usar la videocámara, realizar operaciones complejas con imágenes o crear complejos sistemas de chat.

2.6 METODOLOGÍA UTILIZADA

2.6.1 Scrum

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco

definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Scrum también se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto.

El primer día de la iteración se realiza la reunión de planificación de la iteración. Tiene dos partes:

1. Selección de requisitos (4 horas máximo). El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto. El equipo pregunta al cliente las dudas que surgen y selecciona los requisitos más prioritarios que se compromete a completar en la iteración, de manera que puedan ser entregados si el cliente lo solicita.
2. Planificación de la iteración (4 horas máximo). El equipo elabora la lista de tareas de la iteración necesarias para desarrollar los requisitos a que se ha comprometido. La estimación de esfuerzo se hace de manera conjunta y los miembros del equipo se auto asignan las tareas.

Ejecución de la iteración

Cada día el equipo realiza una reunión de sincronización (15 minutos máximo). Cada miembro del equipo inspecciona el trabajo que el resto está realizando (dependencias entre tareas, progreso hacia el objetivo de la iteración, obstáculos que pueden impedir este objetivo) para poder hacer las adaptaciones necesarias

que permitan cumplir con el compromiso adquirido. En la reunión cada miembro del equipo responde a tres preguntas:

- ¿Qué he hecho desde la última reunión de sincronización?
- ¿Qué voy a hacer a partir de este momento?
- ¿Qué impedimentos tengo o voy a tener?

Durante la iteración el Facilitador (Scrum Master) se encarga de que el equipo pueda cumplir con su compromiso y de que no se merme su productividad.

- Elimina los obstáculos que el equipo no puede resolver por sí mismo.
- Protege al equipo de interrupciones externas que puedan afectar su compromiso o su productividad.

Durante la iteración, el cliente junto con el equipo refinan la lista de requisitos (para prepararlos para las siguientes iteraciones) y, si es necesario, cambian o planifican los objetivos del proyecto para maximizar la utilidad de lo que se desarrolla y el retorno de inversión.

Inspección y adaptación

El último día de la iteración se realiza la reunión de revisión de la iteración. Tiene dos partes:

1. Demostración (4 horas máximo). El equipo presenta al cliente los requisitos completados en la iteración, en forma de incremento de producto preparado para ser entregado con el mínimo esfuerzo. En función de los resultados mostrados y de los cambios que haya habido en el contexto del proyecto, el cliente realiza las adaptaciones necesarias de manera objetiva, ya desde la primera iteración, replanificando el proyecto.
2. Retrospectiva (4 horas máximo). El equipo analiza cómo ha sido su manera de trabajar y cuáles son los problemas que podrían impedirle progresar

adecuadamente, mejorando de manera continua su productividad. El Facilitador se encargará de ir eliminando los obstáculos identificados.

CAPÍTULO III

DESCRIPCIÓN DEL EQUIPO

3.1 Descripción del hardware

Para la implementación del Lazarillo electrónico, se requirió de diversos componentes electrónicos, los cuales al interconectarse logran emular algunos aspectos de utilidad que ofrece un perro lazarillo .

Cabe destacar que el equipo es un prototipo, por lo cual, los componentes utilizados no cumplen con los posibles establecidos para comercializar.

Los componentes utilizados son los siguientes:

- Dispositivo móvil con sistema operativo Android.
- Módulo Bluetooth.
- Módulo controlador
- 1 Cámara digital
- 1 Puente H doble
- 2 Motores
- 2 Ruedas
- 1 Convertidor DC/DC
- 2 Baterias

Estos componentes cumplen con las características necesarias para el desarrollo del proyecto, tanto en lo actualización que posee en dispositivo móvil, como la placa de desarrollo utilizada. Las características de cada uno de estos equipos y componentes se presentan seguidamente, la unión de estos para la implementación del lazarillo se observa en la Figura 10.

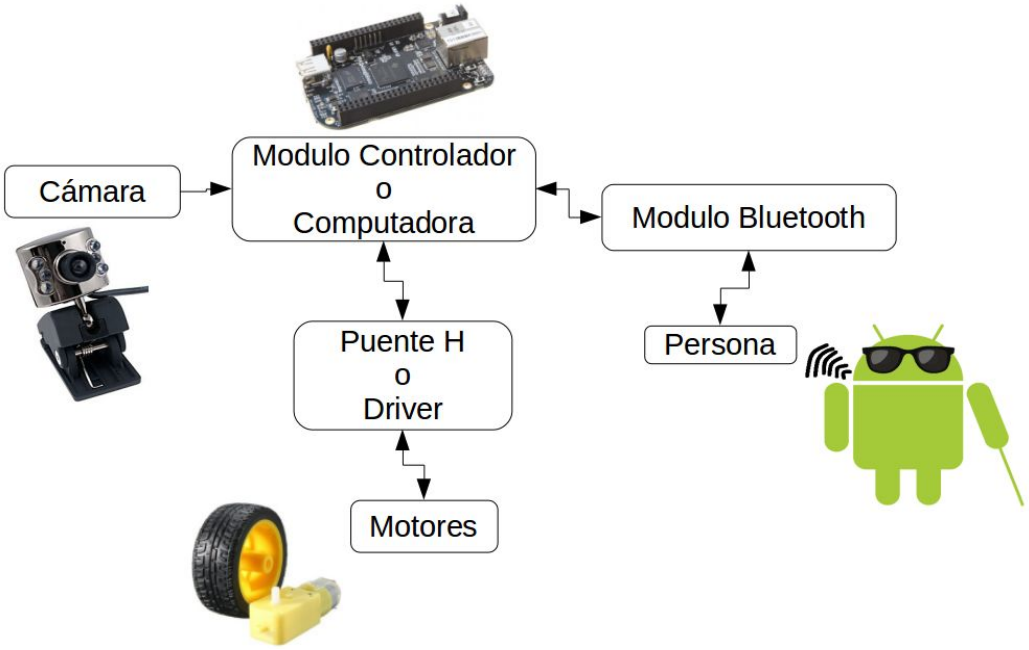

Figura 10: Esquema del equipo.

El dispositivo android utilizado, tiene que presentar características de acuerdo a la aplicación que se desarrolló, esta tiene unos requerimientos que pueden satisfacer los dispositivos android con una de versión de Cupcake 1.5 y un nivel de API de 3 en adelante, las cuales usan actualmente versiones 8.0 y niveles de API de 24-25, de acuerdo a la tabla 12, por lo que prácticamente, cualquier dispositivo android comercial en la actualidad posee la capacidad de correr esta aplicación desarrollada.

El dispositivo android que se utilizó fue un Huawei Ascend P9, el cual es de versión Jelly Bean 4.1-4.3.1 y niveles de API 16-18, este dispositivo se observa en la Figura 11 y tiene las siguientes características.

Figura 11: Dispositivo Android utilizado.

Tabla 3: Características del Dispositivo Android.

Medidas	Largo 132.65mm Ancho 65.5mm Grosor 6.18mm Peso aproximadamente 120g (con batería)
Pantalla	Pantalla : 4.7 pulgadas
CPU	1.5 GHz quad-core CPU K3V2E
Sistema Operativo	AndroidTM 4.2.2
Memoria	RAM : 2GB ROM : 8GB
Network	WCDMA 850/900/1700/1900/2100 MHz GSM 850/900/1800/1900 MHz
GPS	GPS y GLONASS
Conectividad	Wi-Fi : 802.11 b/g/n Bluetooth : Bluetooth 3.0 USB 2.0 with 480 Mbit/s
Sensores	Sensor de Luz Tilt sensor Sensor de proximidad Gyroscope Compass

Cámara	Cámara Principal Resolución: 8.0-megapixel Fotos: 3264 x 2448 pixels Videos: 1920 x 1080 pixels (1080p) Camera Frontal Camera resolución: 5.0-megapixel Fotos: 2592 x 1952 pixels Videos: 1280 x 720 pixels (720p)
Audio	Audio: MP3, WMA 2–9, AMR-NB, AMR-WB, AAC, AAC+, eAAC+, MIDI Formatos de audio: MP3, MP4, 3GP, WMA, OGG, AMR, AAC, FLAC, WAV, MIDI
Video	Grabación: MPEG-4, H.264, H.263, Real Video 7–10, WMV 9, XVID, AC3 Formatos: 3GP, MP4, WMV, RM, RMVB
Emotion UI	Emotion UI (Interfaz visual)
Batería	Capacidad: 2000 mAh Tiempo de carga: menor a 4h

De estas características, las más relevantes para la aplicación desarrollada son la posibilidad de utilizar bluetooth, y el Sistema Operativo, en este caso AndroidTM 4.2.2 ocupando las actualizaciones Jelly Bean de android y niveles de API de acuerdo a la Tabla 2, el cual es apto para utilizar la aplicación realizada.

De acuerdo a la evolución de los dispositivos android (Tabla 2), desde la versión Cupcake de android se tienen las características necesarias para utilizar esta aplicación.

El módulo bluetooth que se utilizó fue el HC-05 ya que de acuerdo a los requerimientos, la persona invidente estará a menos de 2 metros del lazarillo electrónico y este, de acuerdo a sus características, tiene un alcance de 5 a 10 metros, siendo estos más de los necesarios. La posibilidad de implementar un módulo WiFi se encontraba abierta para este proyecto pero, al no tener dicho módulo al alcance, se limitó su empleo. Además, la seguridad que representa la utilización de un módulo bluetooth la hace mas apropiada para los requerimientos del equipo ya que evita el fácil acceso a este por la configuración establecida en los comandos AT, haciendo más segura su utilización.

Figura 12: Módulo Bluetooth HC-05

En la figura 12 se observa la apariencia física del componente y sus respectivas características, de acuerdo a su respectiva hoja de datos, se ven en la tabla 4:

Tabla 4: Características del Módulo Bluetooth HC-05.

Características	**Especificaciones**
Tension de Alimentacion	3,3-6 VDC
Baud Rate Ajustable	1200, 2400, 4800, 9600, 19200, 38400, 57600 y 115200
Corriente de Operación	Menor a 40 mA
Tension de Operacion	3,3 VDC
Alcance	10m
Temperatura de operación	-20 ºC a +75 ºC

El módulo controlador seleccionado fue la tarjeta de desarrollo BeagleBone Black modelo Rev C (Figura 13), ya que cumple con los requerimientos mínimos para el diseño de un prototipo y es la mayor avanzada a nuestra disposición. La tarjeta de desarrollo en cuestion tiene las especificaciones mostradas en la tabla 5.

Nota: Cabe destacar que al momento de realizar las pruebas finales del Lazarillo Electrónico, la placa BeagleBone Black presento averías irreparables. Como sustituto se utilizó un Raspberry Pi 3 para el producto final, debido a que

esta tarjeta es la única a nuestra disposición y se nos hace imposible la adquisición monetaria para obtener una nueva. La programación, características y desarrollo en esta placa es similar en ambos microsistemas.

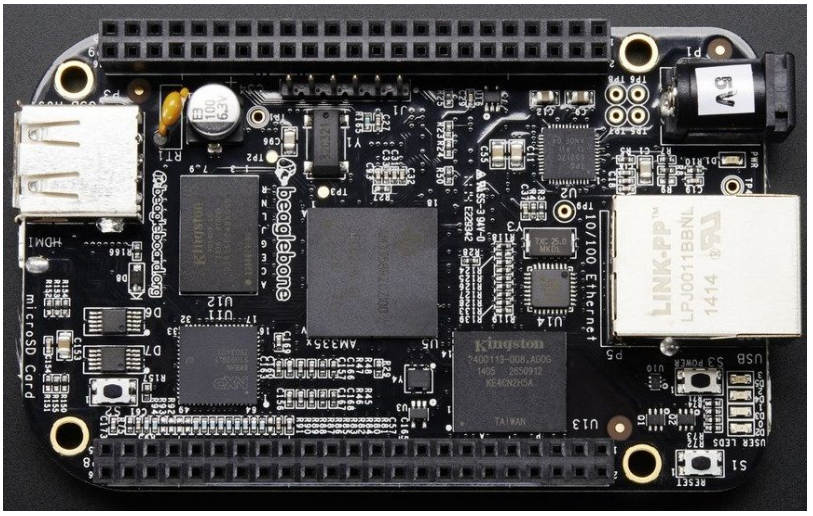

Figura 13: BeagleBone Black Rev C.

Tabla 5: Especificaciones BeagleBone Black Rev C.

Componente	**Especificaciones**
CPU	TI AM335x1GHz Cortex-A8
RAM	512MB DDR3
Puertos USB 2.0	1 puerto
Almacenamiento Interno	4GB 8-bit eMMC
Almacenamiento Externo	Memoria MicroSD-HC. 16GB
Comunicación	Micro USB y/o Ethernet
Fuente de Alimentación	5V/2A
Sistema Operativo	Debian Jessie

La cámara seleccionada fue una webcam USB de 2MP, véase la figura 14, esta no requiere instalación de drivers para su uso en los sistemas operativos, presenta las características de acuerdo a su fabricante en la tabla 6:

Figura 14 : Cámara Digital.

Tabla 6: Especificaciones de cámara USB.

Características	Especificaciones
Fabricante	TOOGOO
Modelo	LE 20.0
Resoluciones Soportadas	640x480, 352x288, 320x240, 176x144, 160x120
Funciones especiales	Micrófono incorporado y 6 LEDs alto brillo

Por otra parte, el Driver utilizado como controlador de corriente para los motores se encuentra en una tarjeta de PCB mostrada en la figura 15 con las características mostradas en la tabla 7

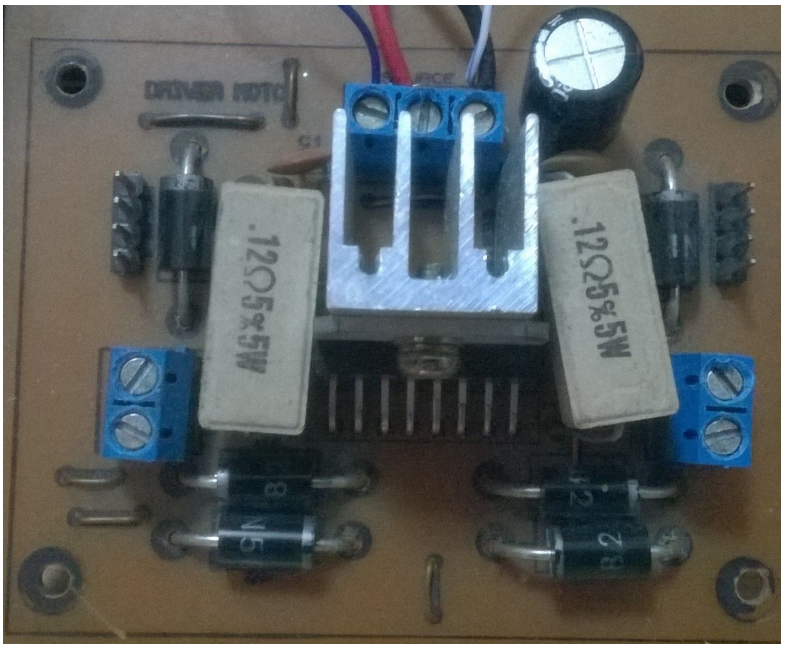

Figura 15: PCB del Controlador.

Tabla 7: Características del Controlador de motores.

Elementos	**Especificaciones**
L298N	Puente-H Doble
8 Pines	Entradas de Control de ambos Puentes
8 Diodos 1N582	Diodos de protección
2 Resistencias 0,12Ω 5%5W	No disponible
3 Condensadores 10pF	Filtros para altas frecuencias
1 Conector Triple	Alimentación y Referencia
2 Conectores Doble	Salidas para cada Motor
Disipador de Calor	No Disponible

Como chasis para el lazarillo electrónico se utilizó una caja de plástico hueca de color Gris, con dimensiones de 15 cm de Largo, 8,5 cm de Ancho y 5 cm de

Alto, ocupando un volumen total de aproximadamente **637,5** cm^3 . Sujeto a este chasis se encuentran dos ruedas con sus respectivos motores eléctricos y una tercera rueda que podrá girar 360° libremente sobre la cual reposara la parte trasera del Lazarillo. Los motores y ruedas a utilizar se pueden observar en la siguiente figura (Figura 16) y sus características en la tabla 8:

Figura 16: Motor con Rueda.

Tabla 8: Características de los Motores.

Características	**Especificaciones**
Tipo de Motor	DC
Tension de Operacion	3-12 V
Velocidad	80-300 rpm
Torque Máximo	2 $\frac{Kg}{cm}$
Consumo	80-100 mA
Tamaño de la rueda	65x26mm

Para energizar al prototipo se utilizaron 2 baterias. Una de ellas será dispuesta únicamente para el consumo de los motores eléctricos y la segunda se utilizará para la Alimentación del BeagleBone Black, en donde a su vez este último permitirá energizar el resto de los módulos como se muestra en la siguiente tabla:.

Tabla 9: Características de las Baterías.

Batería	Especificaciones
Motores	Ni-Cd 9,6V-600mA
BeagleBone Black	Li-Po 7.3V-2200mA

Para la correcta alimentación y funcionamiento del BeagleBone Black, se utilizó un Reductor/Elevador de Tensión para la batería de Li-Po que lo energiza. De este convertidor DC/DC que se muestra en la figura 17, con las características mostradas en la tabla 10

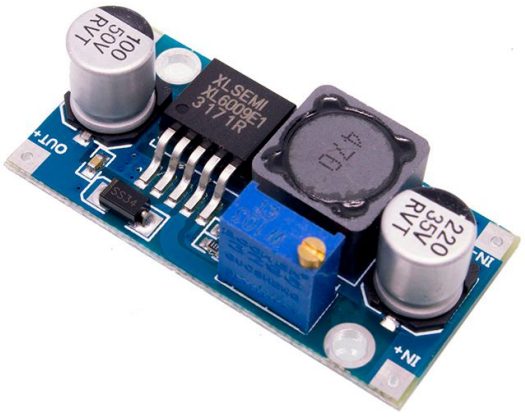

Figura 17: Convertidor DC/DC Boost XL6009

Tabla 10: Características del DC/DC.

Características	Especificaciones
Modelo	Boost XL6009
Tensión de entrada	3V-32V
Tensión de salido	5V-35V

Frecuencia de Operación	400KHz

3.2 Descripción del software

Para el resultado final del proyecto, se desarrolló el software que será detallado a continuación:

- **Android Studio:** Es un entorno de desarrollo integrado oficial para la plataforma Android, la instalación es gratuita y viene integrado con todas las actualizaciones necesarias para el uso de sus librerías con el fin de realizar aplicaciones diversas en la plataforma Android.
- **Java:** Lenguaje de programación de propósito general, concurrente y orientado a objetos con la finalidad de tener la menor dependencia de implementación posible para su reutilización en cualquier dispositivo. Java es el lenguaje utilizado de forma predeterminada en Android Studio.
- **SpeechRecognizer:** Es una librería incorporada en Android Studio para los dispositivos añadido para los niveles API 8 en adelante. Su finalidad es regresar valores String de grabaciones de voz. Esta librería requiere permisos de internet para su utilización,
- **TextToSpeech:** Es una librería incorporada en Android Studio para los dispositivos añadido para los niveles API 4 en adelante. Su finalidad es la de utilizar el software principal del equipo para obtener las voces pregrabadas en este y utilizarla para comunicar por voz lo escrito en String.
- **Bluetooth:** Librería de Android Studio para establecer comunicación mediante bluetooth entre varios dispositivos de acuerdo a los permisos requeridos para su utilización como transmisor o recepción. Su última actualización es para niveles superiores a API 18.

- **OpenCV:** Es una Biblioteca libre multiplataforma que contiene una gran variedad de librerías y funciones respectivas, que a través de la misma se realizará el procesamiento de Imágenes. Versión: 2.
- **Numpy:** Numpy es un módulo básico de Python, el cual ofrece funciones para la resolución de operaciones matemáticas, de esta manera, el proceso de manipulación de la imagen a través de operaciones matriciales se hace más práctico. Version: 1.8.2.
- **Python:** Lenguaje de alto nivel utilizado por su facilidad de uso y comprensión, actualmente el más utilizado en proyectos realizados en placas de desarrollo.

3.3 Descripción de la rutina de reconocimiento de objetivos.

3.3.1. Detección de Colores.

La manera a través de la cual el Lazarillo Electrónico identificó los objetivos para el discapacitado, dentro del contorno cerrado, se realizó a través de la detección de colores. Estos colores o figuras de colores, se encuentran en la parte inferior de cada uno de los accesos o puertas que representarán el lugar al cual el discapacitado indico como destino.

La rutina de detección de colores presenta las siguientes etapas:

1. Lectura de la imagen.
2. Conversión BGR→HSV de la imagen.
3. Binarización de la imagen para un rango de valores según sea el color.
4. Erosión y dilatación de la imagen.
5. Detección de contornos en la imagen.
6. Delineamiento rectangular por cada contorno de la imagen.

Para la captura de imágenes, se trabajó realmente con una secuencia de imágenes, cuya frecuencia de toma de muestras a través de la cámara es ajustable

de acuerdo a la velocidad de respuesta por parte del prototipo. Para la captura de un video o la serie de cuadros por segundo, se necesita crear un objeto de VideoCaptura, cuyo argumento es especificado por el dispositivo o cámara que se desea utilizar.

Una vez que es inicializado este objeto, es posible comenzar a realizar las lecturas de los cuadros en donde para cada uno de ellos se realizará el procesamiento de imágenes.

Las imágenes obtenidas provienen en su espacio de color original, el RGB. Existen más de 150 métodos de conversión entre espacios de colores para imágenes disponibles en OpenCV, pero para efectos de este trabajo especial de grado, solamente se utilizaron los dos más ampliamente usados: BGR↔HSV y BGR↔Gris.

Para la detección o rastreo de colores se trabajó en un espacio de color HSV, es decir, las imágenes de lectura fueron procesadas de BGR→HSV. En el espacio de color HSV, es más fácil representar un color respecto al espacio de color RGB.

Una vez obtenida la imagen original en su espacio de color modificado, esta se hará pasar a través de un método de segmentación de imágenes en HSV para un rango de colores. Este rango de colores utilizado variará de acuerdo al color a detectar, definiéndose mediante un arreglo de 3 dimensiones por medio de la librería Numpy, en la siguiente tabla 11 se pueden observar el rango según sea el color:

Tabla 11: Rango de colores HSV y BGR.

Rango de colores en HSV	Rango de colores en BGR
Rojo: rojo_bajo=array([0,100,60]) rojo_alto=array([5,255,255])	Rojo: rojo_bajo=array([35,35,60])

	rojo_alto=array([0,43,25 5])
Verde: verde_bajo=array([48,40,40]) verde_alto=array([60,255,255])	Verde: verde_bajo=array([34,40,36]) verde_alto=array([0,255,0])
Azul: Azul_bajo=array([110,65,65]) Azul_alto=array([130,255,255])	Azul: Azul_bajo=array([65,54,48]) Azul_alto=array([255,0,85])
Amarillo: Amarillo_bajo=array([16,160,60]) Amarillo_alto=array([25,255,255])	Amarillo: Amarillo_bajo=array([22,42,60]) Amarillo_alto=array([0,213,255])

Una vez procesada la imagen, los valores del color existentes que se encuentren comprendidos entre el rango especificado, serán convertidos en un 1 lógico o Blanco, los que se encuentren fuera del rango en 0 lógico o Negro.

Luego de aplicar este método de segmentación, con la imagen binarizada, se obtiene en blanco nuestro color de interés y en negro lo que no es de interés. En este punto se utilizó una función denominada “Delineamiento Rectangular” o “Bounding Rect” que permite generar un rectángulo que encierra o se genera alrededor de todo el contorno binarizado, obteniendo como resultado las coordenadas en la imagen del punto superior izquierdo del rectángulo (x,y) y su largo y ancho (w,h). Con estas coordenadas y conociendo la resolución de la imagen, es posible conocer la ubicación del punto central del contorno de color detectado en la escena.

De toda la gama de colores existente, se estudiaron y consideraron los que se observan en la Tabla 12.

Tabla 12: Objetivos por color.

Color	Especificaciones
Azul	Destino 1
Verde	Destino 2
Rojo	Destino 3
Amarillo	Destino 4

Principalmente se utilizarán como marcas para los destinos: Rojo, Verde y Azul, esta selección se realizó de esta manera debido a que la distancia entre estos tres colores respecto a su longitud de onda se encuentran lo más equidistantes posible, permitiendo así que en el momento de detección de alguno de estos colores no exista coincidencia alguna con los restantes.

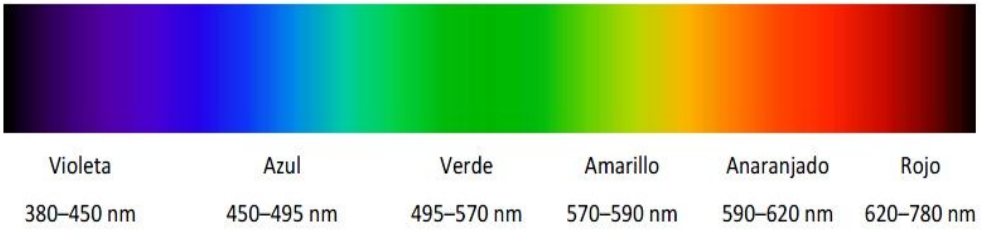

Figura 18: Espectro visible.

Como se puede observar en la figura anterior (Figura 18) las distancias por longitud de onda entre los colores principales (RGB) son aproximadamente de 100nm entre un color y otro.

El diagrama de flujo que representa la rutina que detecta el color en la imagen es el siguiente:

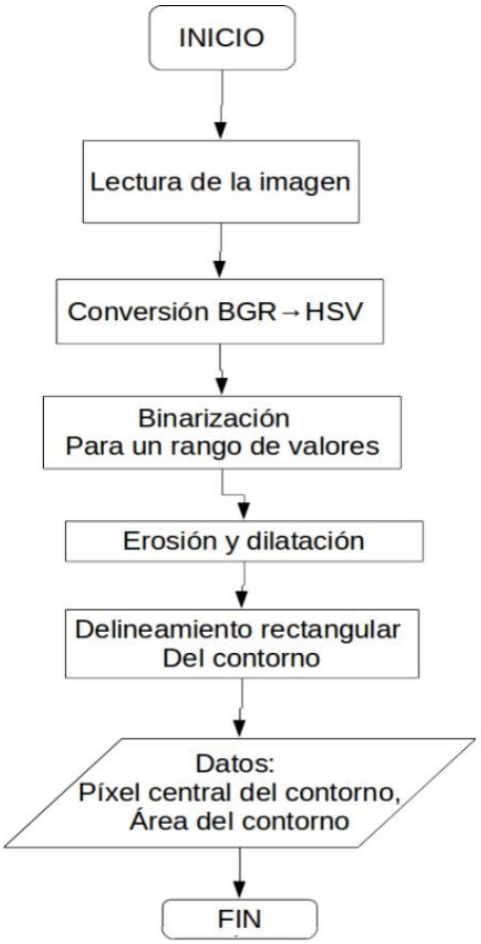

Figura 19: Diagrama de flujo de la rutina de detección de color.

3.3.2. Localización de Colores.

Una vez que el lazarillo electrónico identificó y detectó el color en la imagen, este procederá a localizar la figura en la imagen de acuerdo a la ubicación en la que se encuentre. Solo existen tres posibles ubicaciones del color en la imagen:

-Izquierda: Si la posición del píxel que representa el centro de la figura en la imagen es menor al rango central de píxeles de la imagen.

-Derecha: Si la posición del píxel que representa el centro de la figura en la imagen es mayor al rango central de píxeles de la imagen.

-Centro: Si la posición del píxel que representa el centro de la figura en la imagen se encuentra dentro del rango central de píxeles de la imagen.

Para conocer la localización de los Colores en la imagen, es necesario saber la cantidad de pixeles a lo largo y ancho de ella, de esta manera se puede delimitar las tres zonas o ubicaciones para la imagen. Se utilizó una resolución para la cámara de **320x280**, pudiendo así caracterizar las tres regiones como:

Tabla 13: Zonas en la imagen.

Ubicación	**Rango de Píxeles**
Izquierda	Pixel<120
Derecha	Pixel>200
Centro	120<Pixel<200

Una vez detectado y localizado el color en la imagen, el Lazarillo Electrónico es capaz de accionar sus motores para dirigirse hacia el objetivo que se le indicó.

En las figuras 20, 21 y 22 se pueden observar ejemplos de la detección y localización para el color azul en una imagen:

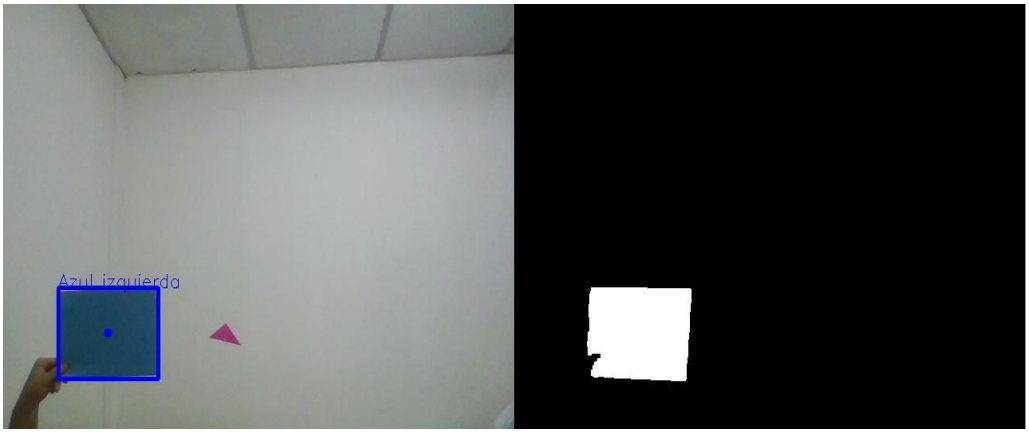

Figura 20: Color Azul a la Izquierda.

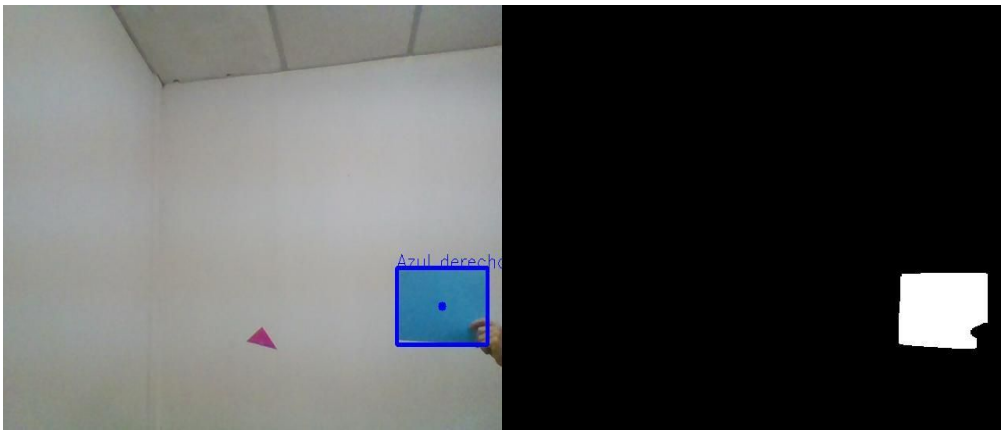

Figura 21: Color Azul a la Derecha.

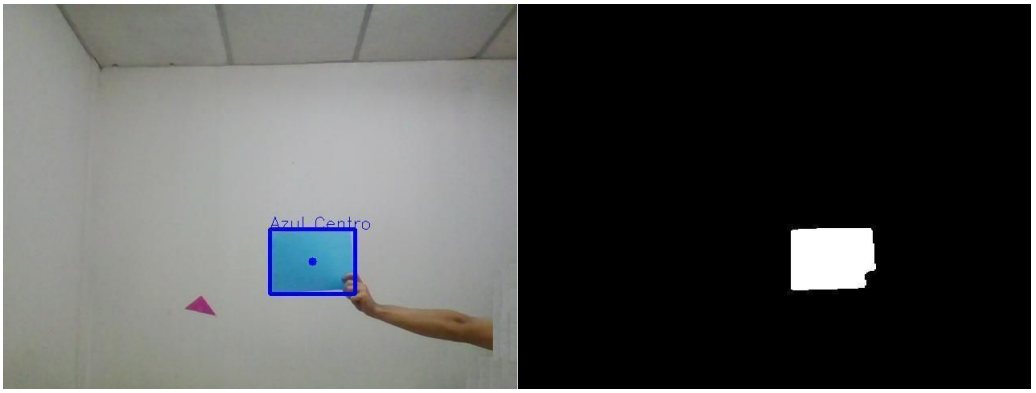

Figura 22: Color Azul en el centro.

A continuación, en la figura 23 se presenta el diagrama de flujo de la localización del color en la imagen:

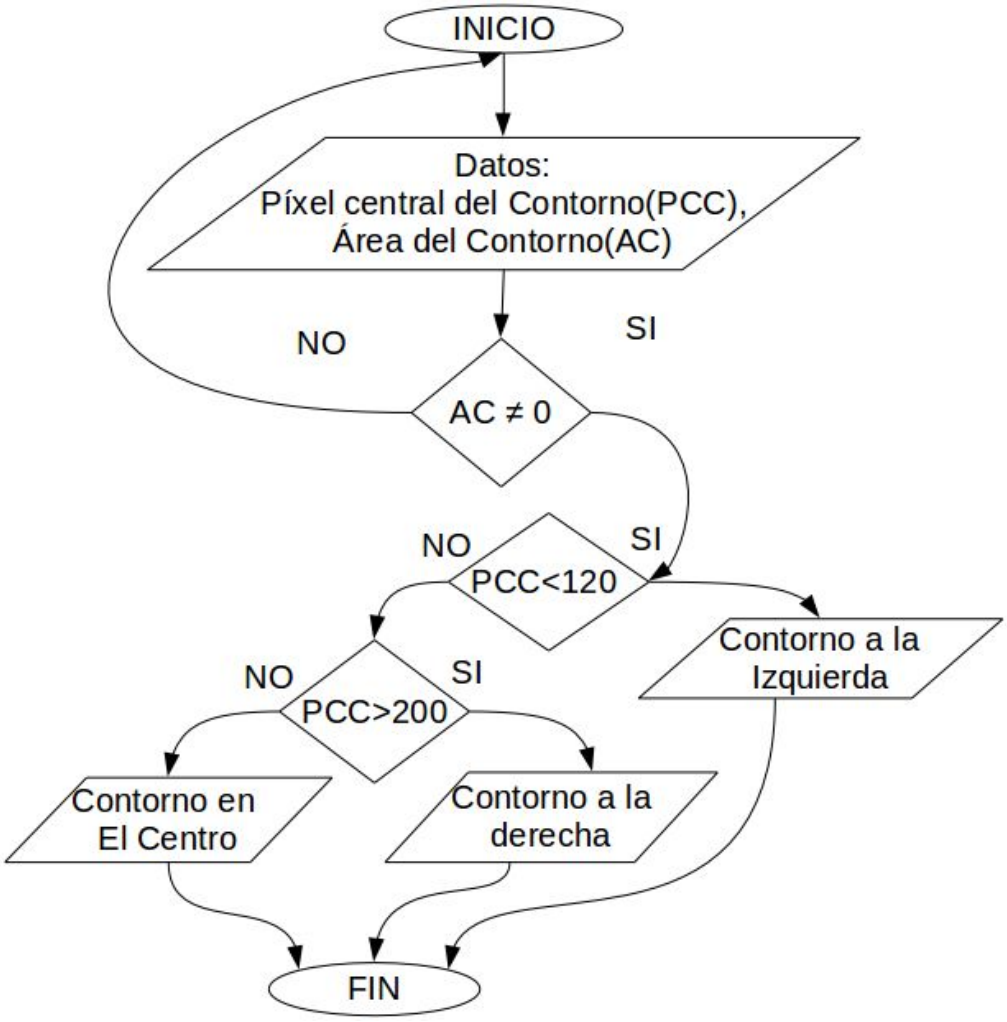

Figura 23: Diagrama de flujo de la localización de colores.

3.4 Descripción de la aplicación móvil

3.4.1 Descripción del sistema

La aplicación móvil abarca la interrelación entre las librerías de reconocimiento de voz, de texto a voz y bluetooth de tal forma que den una interacción con la persona y el lazarillo electrónico de forma amena y sencilla mediante una conversación entre estas.

La aplicación se presenta en una capa, como se muestra en la figura 24 , donde se aprecia con un botón grande el cual fue establecido con la finalidad de que la persona invidente presionando en cualquier lugar al sujetar el celular de la pantalla llegue al botón que tiene como principal objetivo el reconocimiento de voz que indicará a la persona con un sonido característico el inicio del reconocimiento de voz.

Figura 24: Capa de la aplicación móvil.

Esta librería de reconocimiento de voz (Speech to text) consiste en establecer una conexión a los servidores de los APIs de Google para la obtención de un resultado en String del audio grabado, por lo que se requiere acceso a internet.

Se debe tener en cuenta que la aplicación debe tener RECORD_AUDIO, que es un permiso para utilizar esta clase. También requiere permisos de internet del dispositivo y el ancho de banda que consume no es significativo, ya que sólo realiza envíos de palabras cortas con esta aplicación, lo cual no representa un problema en la actualidad.

El hecho de que requiera ancho de banda es que el dispositivo genera una grabación de lo que la persona dice y lo envía a los servidores de Google, el cual retorna una lista de Strings de los resultados más parecidos a los de la grabación.

Esta API de Google es, actualmente, la más acertada en los resultados de reconocimientos de voz, siendo el mismo que tiene el servidor de búsqueda de Google al realizar una búsqueda normal desde internet. De acuerdo a las palabras utilizadas para este proyecto que se encuentran en la Tabla 14 los resultados, indiferentemente de la persona y la pronunciación, tiene un 99% de eficiencia de acuerdo a los resultados arrojados. Como es de esperarse, se debe establecer un idioma para realizar esta búsqueda, la cual se ajusta al idioma que posee el dispositivo móvil que se está utilizando. En otras palabras, si el dispositivo móvil tiene como idioma predeterminado el inglés, las voces en español que se realizan no serán efectivas. Esto se resuelve simplemente cambiando el idioma del dispositivo.

Para la utilización de esta librería, se debe tener en cuenta de forma principal la petición de los permisos en el archivo Manifest de Android Studio para la utilización del ancho de banda del dispositivo. De acuerdo a la Figura 24 se observa que la persona invidente debe presionar un botón para el inicio del grabador de voz, este botón abarca gran espacio de la aplicación para que la persona invidente no tenga problemas al utilizarla.

Cabe destacar que la aplicación se pudo realizar con un reconocimiento de voz continuo, pero este sería propenso a tener errores y generar desorientación a la persona, ya que la aplicación no indicaría en qué momento la persona puede dar inicio a los comandos o el estado en que se encuentra el sistema. Por esta razón, se decidió el uso del botón, ya que este indica a la persona el momento que empieza el reconocimiento con un tono de inicio.

De acuerdo a la entrevista realizada en Asociación de ciegos ubicado en las Acacias a su representante Otto Tovar, un invidente no presenta dificultades para

la utilización de este tipo de aplicaciones móviles, al generar un sonido de inicio de reconocimiento (como se ve en la Figura 25), ubica a la persona el momento para hablar y dar la instrucción deseada.

Figura 25: Capa de reconocimiento de voz

Luego de realizar la grabación, el dispositivo móvil envía esta información a Google para recibir una lista con los principales valores de resultado el cual, en la programación realizada en Java, se toman los dos primeros valores en caso de errores para asegurar que el resultado sea adecuado y se realice la acción. Estas palabras son guardadas en el programa y se procede al envío de un String específico mediante bluetooth al dispositivo sincronizado. En la Tabla 14 se observa la lista de palabras capaces de reconocer y enviar por bluetooth y un String que sirve como codigo enviado por bluetooth al Lazarillo que es de menor número de bits para que el envío sea rápido, ya que este no requiere el conocimiento de la palabra completa y se logra codificar la información de tal forma que sea procesada en el Lazarillo para su respectiva acción.

Cabe destacar que las palabras que se digan en el grabador de voz, no necesariamente son procesadas, existen dos tipos de comandos: aquellos que se envían por bluetooth para las diversas acciones del Lazarillo y aquellos que se procesan dentro de la misma aplicación para realizar acciones dentro de esta.

En la Tabla 15 se observan las instrucciones que no requieren ser enviadas por Bluetooth, como el resultado de "Ayuda", el cual sirve para conocer las diferentes acciones del lazarillo tal como la explicación del sistema para conocer su funcionamiento, logrando que sea más ameno para la persona adaptarse y conocer las posibilidades del lazarillo electrónico.

Nota: cualquier palabra que no exista en la Tabla 14 o Tabla 15, si bien se obtiene su resultado en String, no es procesada ni realiza acciones sobre el Lazarillo, La aplicación se encarga de decir en este caso al invidente que no es una instrucción válida e invita a la persona a los comandos de ayuda para conocer las palabras del funcionamiento del sistema.

Tabla 14: Instrucciones dadas por voz que llegan al Lazarillo Electrónico

Palabra	**Acción**
Détente	Indica al lazarillo si la persona desea detener el movimiento
Avanza	Indica al lazarillo si la persona desea continuar su recorrido
Activar	Activar el lazarillo electrónico
Búsqueda	Indica al lazarillo iniciar protocolo de búsqueda
Lugar A	Indica que desea moverse al Objetivo A
Lugar B	Indica que desea moverse al Objetivo B

Tabla 15: Instrucciones mediante voz que no requieren ser enviadas al lazarillo electrónico

Palabra	**Acción**

Ayuda	Se explica al usuario los comandos para la utilización del lazarillo electrónico
Lugar	Indica al usuario los lugares disponibles del entorno para ir

Por otro parte, se tiene la utilización de la librería text to speech que consiste en sintetizar el habla a partir del texto para la reproducción inmediata o para crear un archivo de sonido. No requiere permisos especiales para su utilización ya que, al crear el proyecto, se generan los permisos estándares en el Manifest que posee cualquier aplicación.

La utilización de esta librería consiste en la escritura de un String, luego se procesa este en el sistema operativo del equipo móvil para reproducir el audio referente a este String, generando una búsqueda de datos presentes en el sistema y reproduciendolos. Por ejemplo, si la persona escribe un string "Hola", el programa buscará en las voces grabadas del sistema esta palabra y la dirá con una voz predeterminada del sistema. El único requisito es la inclusión de la librería al inicio del programa principal además de importarla al inicio del Java.

Como se está trabajando con invidentes, la aplicación mantiene una comunicación constante con la persona mediante strings prescritas en el algoritmo, tanto para orientar a la persona de las posibilidades que posee con el lazarillo electrónico, como para establecer una interacción amena y fácil para la persona, Su utilización se ve a lo largo del programa, para ubicar a la persona en los siguientes momentos:

- Indica a la persona apenas se inicie la aplicación con un mensaje de bienvenida.
- Indica a la persona si se establece o no la conexión de bluetooth con el lazarillo electrónico.
- Informa a la persona invidente de los comandos posibles a utilizar por el lazarillo electrónico.
- En el caso de que el Lazarillo esté realizando una acción y requiera informar de algo a la persona, se envía un dato al celular y este se transforma de un String a

voz indicando a la persona lo que se desee. estos pueden informar si el Lazarillo detecta algo y necesita que la persona se detenga, si el lazarillo está perdido o si ya llegó a su objetivo, tal como se puede ver en la Tabla 17.

Tabla 16: Indicaciones de voz que ofrece la aplicación para ubicar a la persona el estado del proceso

Inicio	Le informa a la persona que la aplicación está abierta y da la recomendación de decir ayuda en caso de no conocer el Lazarillo
Conexión	Indica a la persona si la conexión con el Lazarillo se logró
Protocolo de ayuda	Específica a la persona los comandos de utilidad para el uso del lazarillo electrónico

Tabla 17: Voz que indica resultados obtenidas por el lazarillo electrónico

Détente	Indica que el Lazarillo detectó un obstáculo o la llegara al destino
Perdido	Indica que el Lazarillo no tiene ningún objetivo visible
Lugar A	Indica la llegada al destino A
Lugar B	Indica la llegada al destino B

3.4.2 Descripción del envío de información

Para el envío de información, se utilizó un API de Android Studio llamado Android Bluetooth API. Esta API permite a las aplicaciones conectarse de manera inalámbrica con otros dispositivos bluetooth mediante la comunicación constante de la aplicación y el bluetooth del dispositivo.

Esta API tiene como requerimientos una actualización de versión Android 4.3 (nivel 18 de API) para su utilización, y presenta consumos de energía del dispositivo por la utilización del canal de bluetooth.

Para la utilización de esta API, se deben realizar cuatro tareas para su ejecución, las cuales son:

1. Configurar el Bluetooth.
2. Búsqueda de dispositivos disponibles en el área local.
3. Conectarse a un dispositivo.
4. Transferencia de datos entre dispositivos.

En primer lugar, el dispositivo debe tener la capacidad de soportar Bluetooth para que la aplicación logre la comunicación. Un dispositivo android puede establecer interacción con otros dispositivos android que tengan bluetooth activado, sin embargo, para este proyecto se desea que la comunicación se establezca entre un dispositivo android y un módulo bluetooth, variando los permisos necesarios para establecer esta comunicación.

Esta diferencia de permisos se presenta como un identificador único universal (UUID, por sus siglas en inglés), el cual presenta un formato estandarizado de 128 bits para un ID de string empleado para identificar información de manera exclusiva. Este ID es similar de acuerdo al dispositivo que se busque establecer una comunicación; si se desea establecer una comunicación entre dispositivos móviles, estos tendrán el mismo ID, pero si se desea establecer comunicación con un módulo bluetooth, no se logrará debido a que tienen distintos ID. Es por esto que si se desea establecer una comunicación con un módulo bluetooth, se debe colocar un ID compatible con este tipo de dispositivos. Si se establece una comunicación con un ID de módulo bluetooth, se podrán observar todos los módulos bluetooth adyacentes pero no los dispositivos móviles.

Luego de esta especificación, se logra ver el módulo bluetooth en el dispositivo android, quedando como objetivo siguiente la conexión con este. A fin de crear una conexión estable entre una aplicación y un módulo bluetooth, se debe considerar un BluetoothSocket conectado al mismo canal RFCOMM. En este punto, cada dispositivo puede obtener flujos de entrada y salida, y se puede iniciar la transferencia de datos.

Hasta este punto, se estableció una comunicación con el módulo bluetooth y se logró un intercambio de datos entre el bluetooth del dispositivo móvil y el módulo Bluetooth, pero la información aún no ha sido llevada a la aplicación móvil desarrollada. Esto se hace mediante la verificación del socket, el cual es el vínculo existente entre el bluetooth del dispositivo móvil y la aplicación. Si el socket está habilitado en la aplicación, se mantiene un intercambio de información constante entre lo recibido por bluetooth al dispositivo móvil y la aplicación. Con estos datos se pueden realizar variantes en la programación interna de android studio.

3.5 Configuración del módulo bluetooth

El módulo HC-05 se configura mediante los comandos AT, y estos se habilitan al alimentar con 3,3V el pin “key” del módulo al encenderlo. Los comandos AT permiten cambiar parámetros tales como el nombre del dispositivo, password, modo maestro/esclavo, entre otros.

Para la configuración del módulo, es necesario tener acceso a este mediante una interfaz serial. Debido a esto, se utilizó la plataforma de Arduino ya que posee un IDLE que permite la configuración serial por comandos AT.

Se utilizó el arduino únicamente para la configuración del módulo Bluetooth, debido a su capacidad de realizar una configuración mediante comandos AT.

Cabe destacar que un arduino no presenta los requerimientos mínimos establecidos para la implementación del lazarillo electrónico, por lo cual se utilizó la placa de desarrollo BeagleBone Black, descrito en la sección de hardware del equipo.

Las conexiones necesarias para entrar en los comandos AT desde el IDLE de arduino solo requiere conectar los pines de Tx y Rx de forma invertida en las respectivas Tx y Rx del Arduino utilizado, ya que lo que transmita el Arduino será recibido por el módulo HC-05 para su configuración o acción. Por otra parte, se debe alimentar el módulo y el pin "key" para habilitar su programación mediante los comandos AT.

Hay aspectos que se debieron tomar en cuenta mientras se realizaba esta configuración ya que, al estar utilizando el hardware de UART, la comunicación con la PC no estuvo disponible. Al utilizar el UART para comunicarse con la PC mediante USB, es necesario utilizar un UART emulado por software, de manera que los pines de comunicación con el módulo bluetooth queden en pines distintos, esto se observa en la programación interna del Arduino.

La programación realizada en el IDLE de Arduino, nos permite ingresar a los comandos AT, abriendo una ventana para realizar estos. Esta conexión realizada para acceder a los comandos AT del módulo se observa en la Figura 26.

Figura 26: conexión para establecer los comandos AT

Si bien se pueden configurar de otras formas, ésta es la vía con la que se logró configurarlo ya que no se tenía a disposición un conector USB para el módulo bluetooth.

En la tabla 18 se observan los comandos utilizados para la configuración del módulo HC-05

Tabla 18: Lista de comandos AT utilizados para la configuración del módulo HC-05.

<table>
<tr><td colspan="2">Prueba de funcionamiento:</td></tr>
<tr><td>Enviar: AT</td><td rowspan="2">Al inicio de la conexión, se verifica si el módulo se encuentra conectado y se puede proceder a su programación. para esto se envía el comando AT y se espera su respuesta</td></tr>
<tr><td>Recibe: OK</td></tr>
<tr><td colspan="2">Configurar la velocidad de comunicación:</td></tr>
<tr><td>Enviar:
AT+BAUD<Número></td><td rowspan="2">El módulo HC-05 viene establecido a un Baudrate de 18600 baudios por lo cual se debe cambiar para, tanto la conexión estable con el IDLE de Arduino como al nivel de Baudrate que se requiera, en este caso, fue ajustado a 9600 Baudios.</td></tr>
<tr><td>Recibe: OK<baudrate></td></tr>
<tr><td colspan="2">Configurar el Nombre de dispositivo Bluetooth:</td></tr>
<tr><td>Enviar:
AT+NAME<Nombre></td><td rowspan="2">El cambio del nombre del módulo bluetooth, corresponde al nombre que se verá en los dispositivos que se conecten sobre este.
El nombre ajustado en este proyecto es:
LAZARILLO ELECTRÓNICO</td></tr>
<tr><td>Recibe: OKsetname</td></tr>
<tr><td colspan="2">Configurar el código de vinculación</td></tr>
<tr><td>Enviar: AT+PIN<Pin></td><td rowspan="2">Se configura la contraseña para conectarse a este dispositivo.
En este caso, se usó 1234</td></tr>
<tr><td>Respuesta: OKsetPIN</td></tr>
</table>

3.6 Descripción de la rutina de movimiento

La rutina de movimiento se encuentra conformada por las siguientes funciones:

1. Inicialización.
2. Avanzar.
3. Retroceder.
4. Izquierda.
5. Derecha.
6. Detener.
7. Apagar.

Para el accionamiento de los dos motores Eléctricos, los cuales permitirán el movimiento del lazarillo electrónico en 4 grados de libertad con las funciones antes mencionadas, se utilizaron las librerías disponibles en Adafruit del PWM y GPIO en el BeagleBone Black, de estas librerías también se utilizó el UART's para la parte de transmisión/recepción de información entre el dispositivo móvil y el prototipo,

Para el uso de las rutinas antes mencionadas, se utilizaron las siguientes funcionalidades a nivel de Hardware del BeagleBone Black:

Tabla 19: Herramientas utilizadas del BeagleBone Black.

Funcionalidad	Cantidad	Características	Descripción
PWM	PWM1A PWM1B PWM2A PWM2B	Ciclo Útil: 50% Frecuencia: 1kHz.	Controlar la velocidad de cada motor.
GPIO	GPIO_60 GPIO_44	Salida Lógica de 3,3V.	Habilitar el uso de cada motor.

Las rutinas de movimiento mencionadas anteriormente dependen principalmente de 3 señales de control: 2 PWM y una salida habilitadora (GPIO) por cada motor. Los GPIO cuando se encuentran en Alto, indican la habilitación para el uso del motor respectivo, de lo contrario no será posible poner en funcionamiento dicho motor.

Los movimientos disponibles para el Lazarillo Eléctrico se realizan de acuerdo a las siguientes configuraciones definidas previamente:

Tabla 20: Configuración de los PWM y GPIO para las rutinas de movimiento.

Rutina de Función	Configuración PWM's	Configuración GPIO's
Inicialización	Se habilitan los PWM1 y PWM2 con un Ciclo Útil 0% y frecuencia de señal 1kHz	Se declaran los GPIO's 60 y 44 como salidas Digitales.
Avanzar	Ciclo Útil: PWM1A = 50% PWM1B = 0% PWM2A = 50% PWM2B = 0%	Estado: GPIO_60 = HIGH GPIO_44 = HIGH
Retroceder	Ciclo Útil: PWM1A = 0% PWM1B = 50% PWM2A = 0% PWM2B = 50%	Estado: GPIO_60 = HIGH GPIO_44 = HIGH
Izquierda	Ciclo Útil: PWM1A = 50% PWM1B = 0% PWM2A = 0% PWM2B = 0%	Estado: GPIO_60 = HIGH GPIO_44 = LOW
Derecha	Ciclo Útil: PWM1A = 0% PWM1B = 0% PWM2A = 50% PWM2B = 0%	Estado: GPIO_60 = LOW GPIO_44 = HIGH
Detener	Ciclo Útil: PWM1A = 0% PWM1B = 0%	Estado: GPIO_60 = LOW GPIO_44 = LOW

	PWM2A = 0% PWM2B = 0%	
Apagar	Se deshabilitan los PWM1 y PWM2	No aplica.

Para la velocidad de las ruedas se consideró un Ciclo Útil del 50% con una frecuencia de señal de 1kHz, de esta manera se puede determinar el tiempo de encendido del PWM mediante la siguiente ecuación:

$$D = \frac{\tau}{T} \qquad [1]$$

D : Ciclo Útil 50%.

T : Periodo de la señal 1ms.

τ : Tiempo de encendido 0,5ms.

3.7 Descripción del sistema.

Para el funcionamiento del lazarillo electrónico se tomó en cuenta la aplicación desarrollada para el dispositivo móvil, el reconocimiento de objetivos, la comunicación a través del módulo bluetooth y el control de las rutinas de movimiento, de tal forma que se logre un correcto desempeño de los sistemas en conjunto para un objetivo en común que es la emulación de un aspecto importante del perro lazarillo como lo es llevar a la persona a un lugar de su elección.

Para la implementación de la máquina de estados, se consideró primero las entradas y salidas del sistema que se observan en la tabla 21:

Tabla 21: Entradas y salidas del sistema.

Entradas	**Salidas**
Comandos de voz	Comandos de voz
Cámara	Motores

El proceso del sistema se muestra en la 27 con un Grafcet y la leyenda respectiva a este se muestra en la tabla 22.

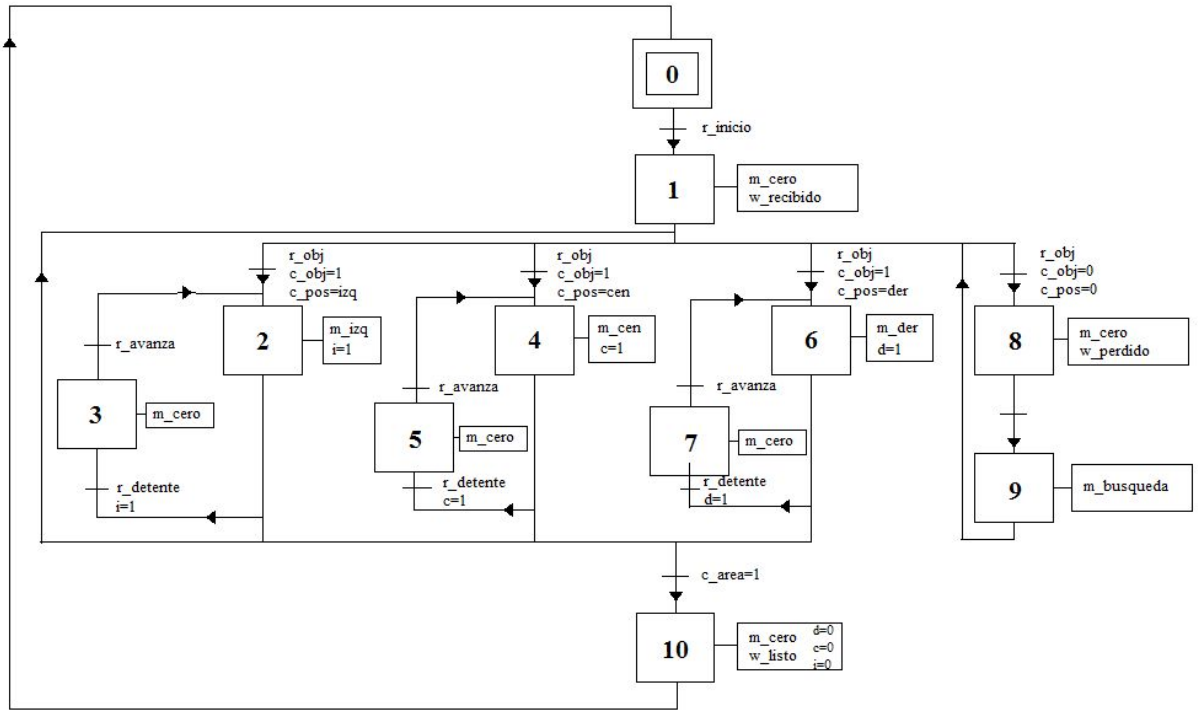

Figura 27: Grafcet del sistema.

Tabla 22: Especificaciones de las entradas y salidas en el Grafcet.

Entrada		**Salida**	
r_inicio	recibe inicio	m_izq	motor izquierda
r_obj	Objetivo seleccionado	m_der	motor derecha
r_detente	détente	m_cen	motor centro
r_avanza	avanza	m_cero	motor detenido
c_obj	objetivo cámara	m_busqueda	búsqueda
c_pos	posición objetivo	w_perdido	envía perdido
c_area	área objetivo	w_listo	envía listo

En el Grafcet mostrado, se observa que el proceso inicia con el recibimiento por parte del celular con la palabra inicio que representa el momento cero. Al recibir esto, el lazarillo pone sus motores cero y envía un mensaje de que se ha iniciado su proceso en el momento 1.

Seguidamente, el lazarillo espera el momento en el que se le asigne un objetivo por el celular y los valores que posee en la entrada de la cámara, objetivo y posición. De acuerdo a los valores obtenidos de la cámara ya se procede a los estados 2, 4, 6 y 8, estas etapas se encargan de generar el movimiento del equipo hasta que, al ver los datos de área de la cámara, sean los adecuados para detenerse e informar la llegada.

Al ingresar en la etapa 2, se empiezan a mover los motores respectivos para ir a la izquierda y se levanta una bandera, esta bandera es utilizada cuando el equipo recibe una instrucción de “detente” por lo que se sigue a la sección 3 y se detienen los motores. Para activar los motores nuevamente, se debe tener como entrada la palabra avanzar y esto hará que el equipo regrese al estado 2.

Si bien el estado 2 asemeja un objetivo posicionado a la izquierda, el estado 4 y 6 también genera movimiento de acuerdo a la posición, en estos casos centro y derecha respectivamente. Estos estados tienen sus respectivas banderas y la posibilidad de detenerse y avanzar.

Como se observa en el grafcet, todas estas etapas están unidas y, en esta unión, se presenta un bucle que consiste en preguntar siempre la posición del objetivo, si el objetivo deja de estar a la izquierda o derecha y pasa a estar en el centro, la verificación con el bucle hace que el sistema se mueva entre los estados 2, 4 y 6.

Por otro lado, si la cámara no logra conseguir el objetivo, el sistema entra en el estado 8, donde se detienen los motores e indica a la persona que no consigue

objetivos a la vista y que procederá a realizar un protocolo de búsqueda mostrado en el estado 9. Este protocolo de búsqueda puede ser llamado indefinidamente hasta lograr encontrar un objetivo. Al observar el objetivo, el programa va directamente al bucle y asigna una dirección de acuerdo al valor de posición que se tiene y se regresa a los estados de direccionamiento del objetivo.

Los estados de direccionamiento 2, 4 y 6 pueden salir del bucle sólo si el objetivo que se busca presenta un área determinada. si el área es mayor a un área en específico, se logra acceder al estado 10, en este, los motores se detienen y se envía a la persona el mensaje de que ya llegó al objetivo.

Finalmente el sistema entra en un bucle final donde se espera la instrucción inicio para volver a iniciar el proceso.

CAPÍTULO IV

PRUEBAS Y RESULTADOS

4.1 Desempeño del sistema

Para la realización de las pruebas del lazarillo electrónico se buscó un espacio abierto y se colocó una lámina de los colores establecidos de tamaño 14X10cm en cada puerta para que estas sean identificadas por la cámara y tratarlas como objetivos o posibles destinos. El suelo en donde se realizaron las pruebas era totalmente plano y sin ningún tipo de irregularidad superficial, con lo cual se asegura un buen desplazamiento del lazarillo.

Con respecto a la aplicación móvil, se ajustó como envío un string de 2 bytes correspondiente por orden de reconocimiento de palabras, esto con la finalidad de distinguir las órdenes y reducir el procesamiento de datos en la placa de desarrollo y así, aumentar la eficiencia.

La primera prueba realizada consistió en colocar el lazarillo de frente a dos posibles objetivos visibles para la cámara. Posteriormente, se le ordenó ir a uno en específico, logrando el direccionamiento hacia ese destino y con dificultades en los controladores de motores.

Para la obtención de un área del contorno deseado lo más exacta posible, fue necesario el cálculo del promedio de una serie de mediciones de esta área, es decir, se leen primero unas aproximadamente 30 fotografías cuyas áreas obtenidas se suman y se promedian al final de las mediciones.

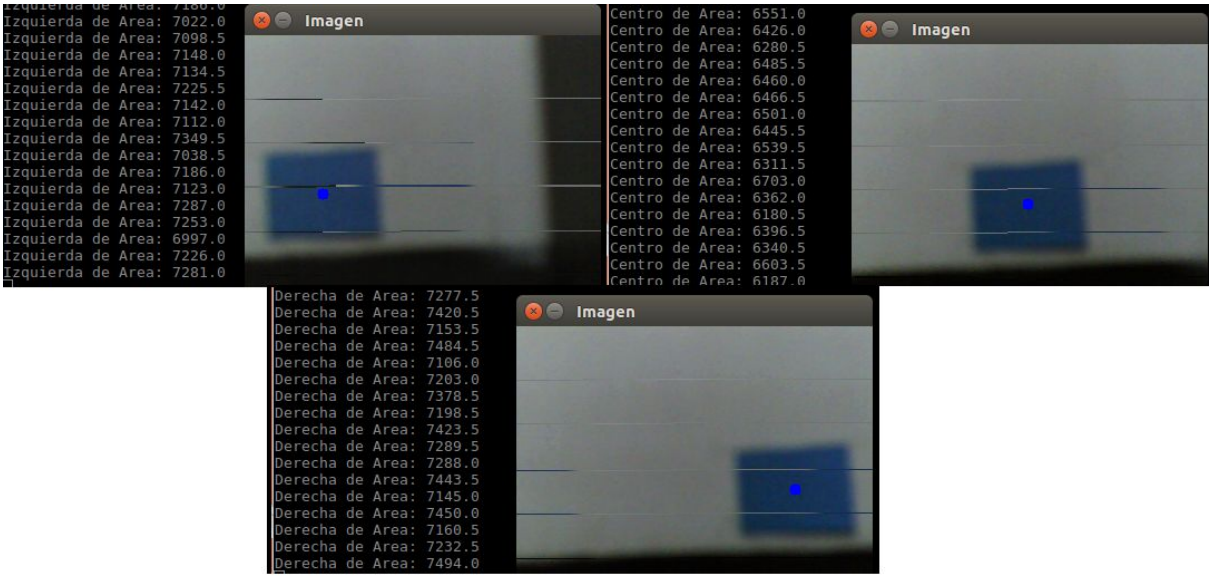

Figura 28: Imagen procesada por el BeagleBone Black vista desde la cámara.

En esta primera prueba se logró determinar que la luminosidad del lugar no es óptima, ya que los valores de reconocimiento del color varían de acuerdo a esta. Esto hace que no se logre detectar el color, por lo cual, el lazarillo no tiene

objetivo al que ir. Esto se logró determinar debido a que se recibía en el celular un mensaje de “perdido” y se detenía el lazarillo por completo.

Estos problemas se explicarán a detalle en la sección 4.1.1.

Para la segunda prueba se aseguró que el entorno tuviera una luminosidad adecuada, de esta forma se evitaron variantes en la programación. Se logró una prueba más efectiva y con tiempos de retardo menores (pero igual notables) debido a la reducción de las exigencias que pedía la cámara al procesador del sistema embebido.

La tercera prueba consistió en probar el comando “perdido” y “búsqueda” del equipo, por lo que se indicó ir a un lugar que no estaba en el entorno. El lazarillo envió la información de que no encuentra el objetivo y realizó el sistema de búsqueda designado, luego se le pidió que realizara el sistema de búsqueda nuevamente pero con el objetivo a la vista y, al realizar el protocolo y ver el objetivo, el lazarillo se dirigió a este sin mayor inconveniente.

4.1.1 Aspectos de la cámara

El número de imágenes o cuadros que procesa la cámara fotográfica influye significativamente en la cantidad de datos y velocidad de respuesta durante el procesamiento de imágenes. Realizando las pruebas se consideró una velocidad de 30 FPS para la cámara, lo cual obtuvo un incorrecto funcionamiento del sistema. Disminuyendo esta velocidad, se pudo lograr un buen funcionamiento del lazarillo, considerando unos 15 FPS.

La luminosidad o intensidad de luz percibida por la cámara es también pilar fundamental para un buen y correcto procesamiento de imágenes. Durante las pruebas se obtuvieron resultados no favorables para espacios con mucha luminosidad.

4.1.2 El entorno con respecto al movimiento

Un inconveniente relevante para el desempeño del lazarillo electrónico es el movimiento que realiza este con respecto al entorno, ya que el lazarillo electrónico reconoce el objetivo pero no reconoce las paredes ni obstáculos en el entorno, si bien el entorno propuesto se encuentra despejado, las paredes terminan siendo un problema con respecto a la posición que tiene el ciego al llegar al objetivo. Para entender mejor este inconveniente, se observa un ejemplo en la Figura 28.

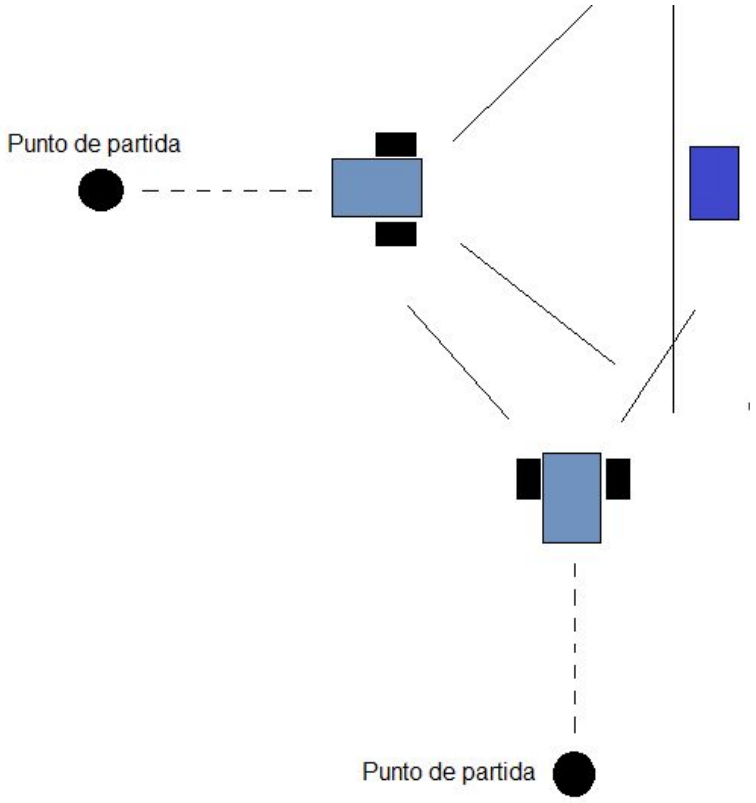

Figura 29: Visión del entorno para el Lazarillo.

En la figura se observa que las posiciones de llegada al objetivo son distintas de acuerdo al lugar de inicio del que se le indicó ir al objetivo, estas diferencias varían el área del color objetivo por las diferentes perspectivas de acuerdo a la posición en la que se encuentra el lazarillo. Cuando el lazarillo se encuentra de frente al objetivo se detiene a una distancia mayor, mientras que desde el punto de

partida inferior el área puede no llegar al valor predeterminado y chocar con la pared.

De acuerdo a la programación realizada, se tiene el aumento del área del objetivo al irse acercando hasta un valor de área específico. Este valor es estático ya que, al no reconocer el entorno y no lograr ver las perspectivas de forma adecuada, siempre tiene un valor específico límite.

CONCLUSIONES Y RECOMENDACIONES

Durante el desarrollo de este Trabajo Especial de Grado se entendió una parte del entorno de las personas discapacitadas visualmente, esto en función de un prototipo no comercial de lazarillo electrónico cuyo diseño final le brinde a dichas personas discapacitadas una tecnología como búsqueda de la Independencia elemental en sus entornos de la vida diaria. De acuerdo a la entrevista con el Director del centro de rehabilitación de clínicas para ciegos, Tovar Otto, normalmente los obstáculos que se le presentan a estas personas discapacitadas son los que se encuentran obstruyendo el paso del Lazarillo en una trayectoria cuyo destino es de su preferencia, teniendo como destinos más comunes: habitaciones, baños, pasillos, cocinas, comedores, etc.

El lazarillo electrónico reconoce correctamente los colores azul, verde y rojo en espacios sin mucha intensidad de luz. El tiempo de respuesta entre el prototipo y los comandos de voz son despreciables. El vehículo prototipo se orienta hacia los objetivos de colores sin inconvenientes.

Fue necesario disminuir la resolución de la cámara de 640x480 a 320x280 para evitar congelamientos del sistema embebido debido a la mayor

cantidad de píxeles que se procesaban, por otro lado también fue necesaria la reducción de los FPS utilizados, siendo 15 en vez de 30, de esta manera mejoraba la velocidad en la respuesta del sistema. Los entornos o áreas cerradas en donde la cantidad de intensidad de Luz es tal que se manifiesta como ruido en la fotografía, suelen arrojar resultados no favorables ocasionando respuestas no deseadas por parte del Lazarillo. Las baterías utilizadas para la alimentación del micro sistema posee una capacidad para energizar por un tiempo de duración no mayor a 20 minutos.

REFERENCIAS BIBLIOGRÁFICAS

[1] Tovar, Otto. *Manual de accesibilidad como diseño y derecho universal para todos.* (Libro). Asociación Civil sin Barreras. 1ra edición 2007, Caracas. Cap 3

[2] Gutiérrez M., Jesús A. *El perro guia,* (Artículo)[En Línea].
<http://www.adiestradorcanino.com/webdelperro/el-perro-guia/142>
[Consulta: Enero, 2017]

[3] Noguer, Arregui. *Tecnología y discapacidad visual : necesidades tecnológicas y aplicaciones en la vida diaria de las personas con ceguera y discapacidad visual.* (Libro). SS Informes. 2004. capítulos 1-4.

[4] Salvador Gómez, Oliver. *Curso programacion android V2.* (Libro). USGY, 2008. capítulo 2.

[5] Terrero, Henry & Paredes José. *Desarrollo de aplicaciones con Java.* (Libro). Editorial Fundación de Código Libre Dominicano, 2010, 1ra edición, capítulo 3.

[6] Gerber, Adam & Craig, Clifton. *Aprende Android Studio: desarrolla aplicaciones android de forma rápida y eficiente,* Apress, 1ra edición, 2015, capitulos 3 y 5.

[7] Acharya, T., Ray, A. K. (2005) *Procesamiento de imagenes: principios y aplicaciones*. John Wiley & Sons.

[8] Gonzalez, Rafael (2008). '*Digital Image processing, 3rd*'. Pearson Hall.

[9] OpenCV Org. *Changing ColorSpaces.* [En línea]. <http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_colorspaces/py_colorspaces.html#converting-colorspaces> [Consulta: Enero 2016]

[10] OpenCV Org. *Image Thresholding.* [En línea]. <http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_thresholding/py_thresholding.html#thresholding> [Consulta: Enero 2016]

[11] OpenCV Org. *Morphological Transformation*. [En línea]. <http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html#morphological-ops> [Enero: 2016]

[12] OpenCV Org. *Contours in OpenCV*. [En línea]. <http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_contours/py_table_of_contents_contours/py_table_of_contents_contours.html#table-of-content-contours> [Consulta: Enero 2016]

[13] Howse Joseph. OpenCV, (Abril 2013) *Computer Vision with Python*, (Libro). Livery Place, 35 Livery Street, Birmingham B3 2PB, UK. Pack Publishing Ltd.

[14] Adafruit. *Using the Adafruit_BBIO Library*. [En línea]. <https://learn.adafruit.com/setting-up-io-python-library-on-beaglebone-black/using-the-bbio-library> [Consulta: Mayo 2015]

ANEXO I

MANUAL DE USUARIO

Para el uso pertinente del sistema se debe tener instalado la aplicación TalkBar de Android que permite el uso de un dispositivo móvil Android mediante el desplazamiento de de las aplicaciones dirigido por voz y moverse por este con solo dos tipos de órdenes sobre el celular:

1. Desplazamiento hacia abajo sobre la pantalla táctil.
2. Desplazar hacia la derecha sobre la pantalla táctil

El desplazamiento hacia abajo sobre la pantalla táctil permite a la persona moverse por las aplicaciones del celular, cada movimiento es indicado a la persona al ser realizado por voz explicando la sección en la que se encuentra la persona de las aplicaciones y características existentes en el celular.

Al desplazar el dedo hacia la derecha sobre la pantalla táctil, permite a la persona dirigirse a el escritorio seleccionado de acuerdo a la sección que indica por voz el programa.

La pantalla principal en este caso se observa en la Figura 1.1, donde se procederá a escuchar cada sección y buscar la aplicación Lazarillo electrónico, para esto debe desplazarse 8 veces hacia abajo se logra acceder a la carpeta de aplicaciones.

Figura 1.1: Imagen del escritorio del celular por Talkbar.

Al presionar y acceder a la sección de aplicaciones, se accede a la sección de aplicaciones del móvil como se muestra en la Figura 1.1, en esta se debe deslizar una vez hacía abajo para que el programa informe que se encuentra sobre la carpeta de aplicaciones, por lo que quedaría deslizar hacia la derecha para moverse por las diferentes aplicaciones del celular. Dichas aplicaciones están organizadas de forma alfabética, así que se desplaza a la derecha hasta que la voz indique que está sobre la aplicación móvil llamada lazarillo electrónico, al observar en la pantalla se verá como en la Figura 1.2.

Figura 1.2: Sección de aplicaciones de talkbar

Figura 1.3: Localización de la aplicación lazarillo electrónico

Al escuchar que se está sobre la aplicación de lazarillo electrónico, se desplaza hacia abajo para ejecutar la aplicación seleccionada y finalmente se desplaza a la derecha para ingresar a esta.

Al ingresar a la aplicación, se sabrá que se accedió ya que se emitirá un saludo de bienvenida y se recomendará decir ayuda para ingresar a un protocolo que se encarga de explicar las diferentes órdenes para la utilización del Lazarillo Electrónico. Además, la aplicación tiene una presentación como se muestra en la Figura 1.4.

Figura 1.4: Pantalla principal de la aplicación de lazarillo electrónico.

Inicialmente, se debe establecer la conexión bluetooth con el lazarillo, al realizar esto, las voces resultantes del programa indicará si se encuentra conectado con la aplicación.

Para dar las órdenes se debe presionar el botón send que sale en la pantalla y se escuchará un sonido característico como indicador de que es el momento de hablar y dar una orden. En este momento la pantalla lucirá como en la Figura 1.5.

Figura 1.5: Vista de la aplicación al reconocer la voz.

Para iniciar con el lazarillo electrónico, se dará la indicación "INICIÓ" el cual activara el lazarillo y dará un mensaje de vuelta diciendo que el Lazarillo está listo para su utilización.

Las órdenes para Acción del Lazarillo son las siguientes:

Tabla 1.1: Lista de órdenes para el Lazarillo Electrónico.

Palabra	**Acción**
Détente	Indica al lazarillo si la persona desea detener el movimiento
Avanza	Indica al lazarillo si la persona desea continuar su recorrido
Activar	Activar el lazarillo electrónico
Búsqueda	Indica al lazarillo iniciar protocolo de búsqueda
Lugar A	Indica que desea moverse al Objetivo A
Lugar B	Indica que desea moverse al Objetivo B

Antes de iniciar, se debe tener tener en las manos la correa del lazarillo electrónico. El lazarillo debe ubicarse en la mano izquierda de la persona a medio metro por delante para evitar tropiezos. Se debe tener siempre el bastón a la mano derecha para moverlo y mantener siempre el contacto con el entorno en que se encuentra, para conocerlo con la ayuda del lazarillo y el bastón. La posición que de debe tener es la mostrada en la figura 1.6.

Figura 1.6: Perro Lazarillo guiando a persona discapacitada.

Se debe siempre mantener la posición indicada y enfocarse en reconocer el camino con el bastón, ya que el lazarillo electrónico es una herramienta de aprendizaje para adaptarse a la discapacidad.

Para iniciar un proceso adecuado, se debe ingresar el lugar al que se quiere ir y, ya con el conocimiento por parte del lazarillo, las acciones “detente” y “avanza” se habilitan para su uso ya que el lazarillo estaría en un proceso.

El Lazarillo tiene dos acciones luego de que le den una información:

1. Si ve el objetivo.
2. Si no ve el objetivo.

Si el lazarillo ve el objetivo, indica a la persona que iniciará el recorrido y empezará el movimiento de este hacia el lugar pedido.

Por el contrario, si el lazarillo no ve el objetivo, indica a la persona que iniciará un protocolo de búsqueda, y realiza un escaneo del entorno para buscar el objetivo. Si lo consigue, va a él y si no lo consigue indica a la persona que no reconoce el lugar o que está perdido.

En el caso de que el equipo se encuentre perdido, la persona puede ordenarle nuevamente que realice el protocolo de búsqueda cuando quiera mediante voz y este volverá a escanear el entorno hasta encontrarlo.

ANEXO II

IMÁGENES DEL PROTOTIPO

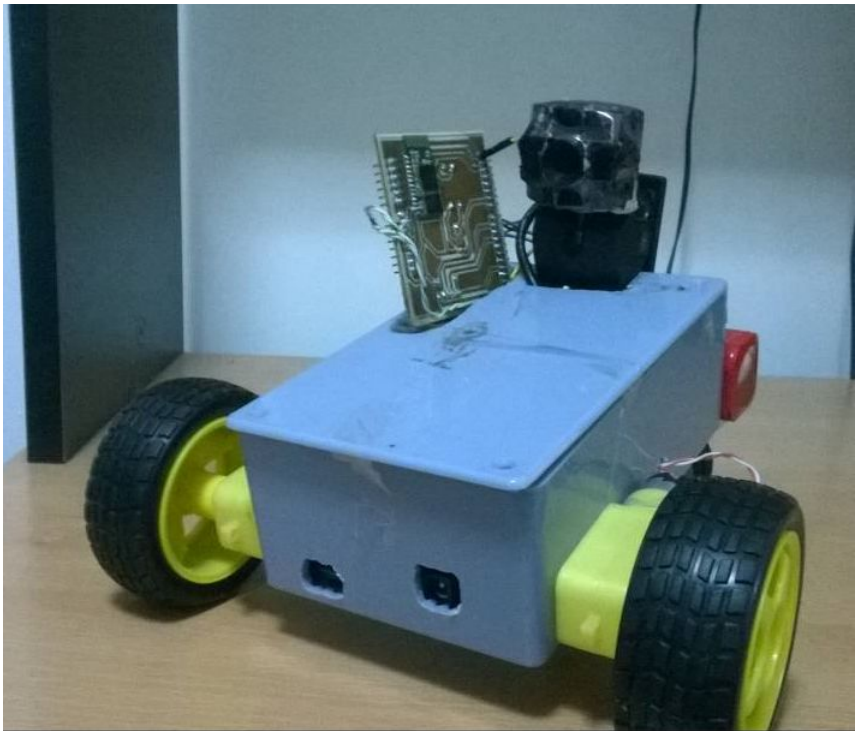

Figura 2.1: Prototipo del lazarillo electrónico

Figura 2.2: Imagen frontal del prototipo del lazarillo electrónico

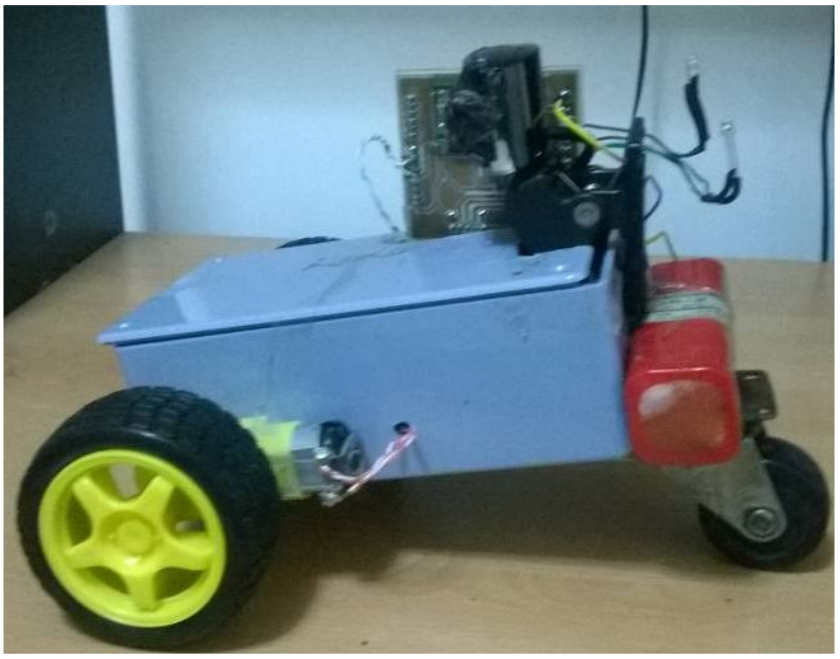

Figura 2.3: Imagen lateral del prototipo del lazarillo electrónico

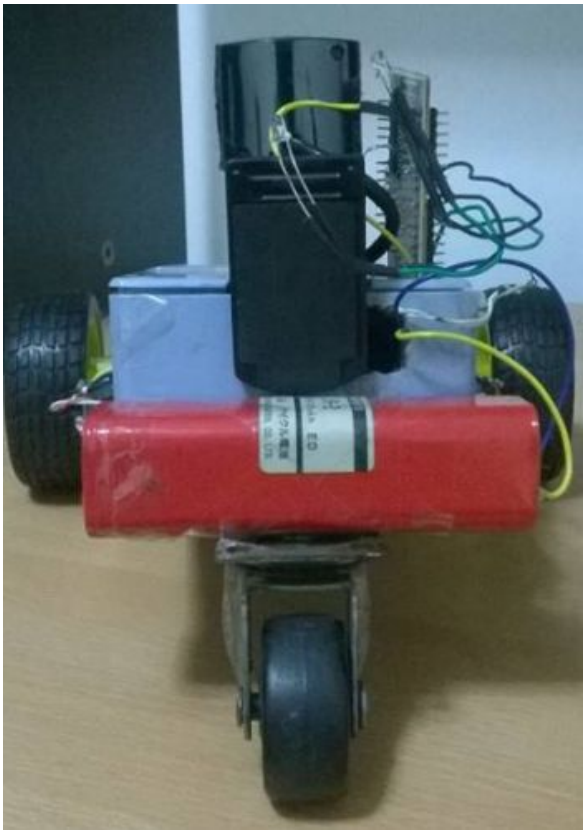

Figura 2.3: Imagen trasera del prototipo del lazarillo electrónico