ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA ELECTRÓNICA Y AUTOMATIZACIÓN

"DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO EMBEBIDO DE RECONOCIMIENTO DE OBJETOS POR MEDIO DE VISIÓN ARTIFICIAL COMO AYUDA A PERSONAS CON DEFICIENCIA VISUAL"

Trabajo de titulación

Tipo: Dispositivo Tecnológico

Presentado para optar al grado académico de:

INGENIERO EN ELECTRÓNICA Y AUTOMATIZACIÓN

AUTORES:

MARÍA FERNANDA GUILLEN SALAZAR

TELMO JAIME BONILLA BONILLA

Riobamba – Ecuador

2021

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA ELECTRÓNICA Y AUTOMATIZACIÓN

"DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO EMBEBIDO DE RECONOCIMIENTO DE OBJETOS POR MEDIO DE VISIÓN ARTIFICIAL COMO AYUDA A PERSONAS CON DEFICIENCIA VISUAL"

Trabajo de titulación

Tipo: Dispositivo tecnológico

Presentado para optar al grado académico de:

INGENIERO EN ELECTRÓNICA Y AUTOMATIZACIÓN

AUTORES: MARÍA FERNANDA GUILLEN SALAZAR

TELMO JAIME BONILLA BONILLA

DIRECTOR: ING. OSWALDO GEOVANNY MARTÍNEZ GUASHIMA

Riobamba – Ecuador

2021

Nosotros, María Fernanda Guillen Salazar y Telmo Jaime Bonilla Bonilla, declaramos que el presente trabajo de titulación es de nuestra autoría y los resultados de este son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autores asumimos la responsabilidad legal y académica de los contenidos de este trabajo de titulación; El patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 23 de diciembre de 2021

María Fernanda Guillen Salazar

060427133-8

Telmo Jaime Bonilla Bonilla

020250043-5

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA ELECTRÓNICA Y AUTOMATIZACIÓN

El Tribunal del Trabajo de Titulación certifica que: El trabajo de titulación; tipo: Dispositivo tecnológico, **"DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO EMBEBIDO DE RECONOCIMIENTO DE OBJETOS POR MEDIO DE VISIÓN ARTIFICIAL COMO AYUDA A PERSONAS CON DEFICIENCIA VISUAL"**, realizado por la señorita **MARÍA FERNANDA GUILLEN SALAZAR** y por el señor **TELMO JAIME BONILLA BONILLA**, ha sido minuciosamente revisado por los Miembros del Trabajo de Titulación, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal Autoriza su presentación.

	FIRMA	FECHA
Ing. Verónica Elizabeth Mora Chunllo **PRESIDENTE DEL TRIBUNAL**	VERONICA ELIZABETH MORA CHUNLLO Digitally signed by VERONICA ELIZABETH MORA CHUNLLO Date: 2022.01.06 07:00:36 -05'00'	2021-12-23
Ing. Oswaldo Geovanny Martínez Guashima **DIRECTOR DE TRABAJO DE TITULACIÓN**	OSWALDO GEOVANNY MARTINEZ GUASHIMA Firmado digitalmente por OSWALDO GEOVANNY MARTINEZ GUASHIMA Fecha: 2022.01.06 08:28:50 -05'00'	2021-12-23
Ing. Mayra Alejandra Pacheco Cunduri **MIEMBRO DEL TRIBUNAL**	Firmado electrónicamente por: MAYRA ALEJANDRA PACHECO CUNDURI	2021-12-23

DEDICATORIA

El desarrollo de este trabajo de titulación está dedicado a Dios por darme salud y vida para que pueda culminar mis estudios. A mi padre Napoleón ya que con su amor ha sido mi apoyo fundamental y mi ejemplo más grande de trabajo y superación a lo largo de toda mi vida, a mi madre Normita que me guía y me brinda sus bendiciones desde el cielo, mis hermanas Anita y Cristina ya que creyeron en mí y han sabido estar en todo momento apoyándome, guiando y aconsejándome, para alcanzar este gran logro. A mis cuñados Rodrigo y Byron ya que con cada una de sus palabras fueron una motivación para alcanzar esta meta. A mi hija Alisson ya que ella es la persona más importante en mi vida la cual ha sido mi motor en cada uno de mis días y a la que amo con todo mi corazón. A mi tía Magolita por estar pendiente de mi a pesar de encontrarse lejos y a mis amigas Alexandra, Sandra y Mariuxi por ser esa amistad sincera y leal la cual han sabido acompañarme en este camino.

María

A Dios, por permitirme llegar a este momento tan especial en mi vida. Por los triunfos y los momentos difíciles que me han enseñado a valorarlo cada día más. A mi madre por ser la persona que me ha acompañado durante todo mi trayecto estudiantil y de vida. A mi padre quien con sus consejos ha sabido guiarme para culminar mi carrera profesional. A mi hermana que siempre ha estado junto a mí y brindándome su apoyo, muchas veces poniéndose en el papel de padre. A mis profesores, gracias por su tiempo, por su apoyo, así como por la sabiduría que me transmitieron en el desarrollo de mi formación profesional.

Telmo

AGRADECIMIENTO

En primer lugar, agradezco a Dios por darme salud y vida para seguir en este camino hasta poder culminar mi meta y ver alcanzado este gran éxito. A mi ejemplo más grande de vida que son mis padres y hermanas, por su amor, su apoyo incondicional y sobre todo por creer en mi para alcanzar este título profesional. A mi hija Alisson por ser mi fuerza para levantarme cada uno de mis días para cumplir este sueño que lo veo hecho realidad, a nuestro tutor de tesis el Ing. Oswaldo Martínez ya que gracias a su guía y ayuda se pudo terminar el trabajo de titulación, al Ing. José Guerra ya que más que un docente ha sido un amigo y un apoyo a lo largo de mi formación profesional, a la Escuela Superior Politécnica de Chimborazo por ser la casa en la que han sido adquiridos cada uno de mis conocimientos, a mis amigas/os y compañeros con los cuales compartí tristezas, alegrías, conocimientos que quedaran grabados en mi memoria siempre y a cada una de las personas que fueron parte de este proceso gracias.

María

En primer lugar, doy infinitamente gracias a Dios, por haberme dado fuerza y valor para culminar esta etapa de mi vida. Agradezco también la confianza y el apoyo brindado por parte de mi madre, que sin duda alguna en el trayecto de mi vida me ha demostrado su amor, corrigiendo mis faltas y celebrando mis triunfos. A mi hermano, que con sus consejos me ha ayudado a afrontar los retos que se me han presentado a lo largo de mi vida. A mi padre, que siempre lo he sentido presente en mi vida.

Telmo

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE GRÁFICOS

ÍNDICE DE ANEXOS

ÍNDICE DE ABREVIATURAS

A:	Amperio
BGR:	Azul, verde, rojo
CA:	Corriente alterna
CD:	Corriente directa
CONADIS:	Consejo Nacional para la Igualdad de Discapacidades
ETHERNET:	Estándar de redes de área local
HW:	Hardware
mm:	Milímetros
OMS:	Organización Mundial de la Salud
RAM:	Memoria de acceso aleatorio
RSPICAM:	Cámara de tarjeta Raspberry Pi
SW:	Software
USB:	Universal Serial Bus
V:	Voltios

RESUMEN

El presente trabajo de titulación se realizó con la finalidad de diseñar y construir un prototipo electrónico para personas con deficiencia visual que sirva como una ayuda al momento de desplazarse en entornos interiores (indoor). Dentro de la problemática del trabajo se da ayuda enel ámbito social e inclusivo en el desenvolvimiento de su vida cotidiana, por medio de la tecnología y la visión artificial. La construcción del prototipo se realizó con elementos como Rasberry Pi 4, cámara Pi Noir, parlantes entre otros. El desarrollo del sistema de detección se basó en el algoritmo "Haar Cascade" y su interfaz fue implementada a través de la interfaz gráficade usuario (GUI) Cascade Trainer. En el primer módulo se realiza la captura de imágenes por medio de la cámara Pi Noir las cuales son procesadas en la tarjeta de desarrollo Rasberry Pi 4, elsegundo módulo realiza una comparación de imágenes, es decir mediante el algoritmo descrito seobtiene el reconocimiento de imágenes con información de nombre, posición, tamaño de objetoen pantalla y distancia, para finalizar el tercer módulo captura toda la información anterior y la envía al sector de conversión a voz y la transforma en un archivo mp3. Los módulos se comunican de manera alámbrica permitiendo al prototipo indicar si existe un objeto cerca del usuario, por loque emite un mensaje de voz con los parámetros descritos, facilitando de esta manera la toma de decisiones para un desplazamiento seguro. Se evaluó el prototipo tomando un sin número de pruebas de los diferentes objetos, de lo cual se obtuvo una media de 87% de efectividad. Se concluye, que es un dispositivo capaz de cumplir con las expectativas establecidas para mejorar el estilo de vida de personas con deficiencia visual.

Palabras clave: <VISIÓN ARTIFICIAL>, <DISCAPACIDAD VISUAL>, <OBJETOS>, <RECONOCIMIENTO DE IMÁGENES>, <PROCESAMIENTO DE SEÑALES>, <SISTEMAS EMBEBIDOS>. <SISTEMAS EMBEBIDOS>.

ABSTRACT

The current graduate research project was to design and build an electronic prototype for people with visual impairments as a support at the time of moving in indoor environments. Within the work problem, help is given in the social and inclusive field in the development of their daily life through technology and artificial vision. The construction of the prototypewas carried out with elements such as Raspberry Pi 4, Pi Noir camera, speakers, among others. The detection system development is based on the "Haar Cascade" algorithm, and its interface was implemented through the Graphic User Interface (GUI) Cascade Trainer. In thefirst module, images are captured by Pi Noir camera, which are processed on the Raspberry Pi 4 development board; the second module performs a comparison of images, that is, through the algorithm described, the image recognition is obtained with the information aboutname, position, object size on screen and distance, to finish the third module captures all theprevious information, and it sends to the speech conversion sector and transforms it into an mp3 file. The Modules can communicate with each other in a wired way allowing the prototype to indicate if there is an object near the user; therefore, the system issues a voice message with the parameters described, thus facilitating the decision-making for safe movement. The prototype was evaluated by taking several tests of the different objects, of which an average of 87% of effectiveness was obtained. It is concluded that the device can meet expectations to improve the lifestyle of people with visual impairments.

Keywords: <ARTIFICIAL VISION>, <VISUAL IMPAIRMENT>, <OBJECTS>,<IMAGE RECOGNITION>, <SIGNAL PROCESSING>, <EMBEDDED SYSTEMS>.

Firmado electrónicamente por:
LENIN
IVAN LARA

INTRODUCCIÓN

De acuerdo con la Organización Mundial de la Salud (OMS), se estima que a nivel mundial 285 millones de personas tienen deficiencias visuales, de las cuales 39 millones tienen ceguera y 246 millones son débiles visuales. De estas, el 90 % viven en países en desarrollo Según la OMS (INFOSALUS, 2016).

La atención a las personas con discapacidad en el Ecuador se ha caracterizado como en los problemas sociales, por ser de baja cobertura y deficiente. Ecuador tiene 14,8% de discapacitados, según OEA en el Ecuador existen más de 200.000 ecuatorianos no videntes y deficientes visuales según el conadis. En el cantón de Riobamba de acuerdo con el conadis existen 769 personas con deficiencia visual (CONADIS, 2021a).

Las personas con ceguera total o con poca visión usualmente tienen problemas para manejarse fuera de entornos conocidos, por esta razón, muchas personas con poca visión caminan junto a un amigo o familiar que los ayude a conducirse en entornos desconocidos .Esto provoca un problema de inseguridad para estas personas, pueden originar accidentes de tránsito (DI GENNARO, 2017).

La tecnología remota de Aíra utiliza lentes inteligentes para conectar a aquellas personas con visión reducida a una red de agentes certificados. Los agentes pueden "ver" desde la perspectiva del usuario utilizando video en casi tiempo real y luego comunicarle información e instrucciones al usuario. Esto ayuda a que el usuario logre hacer tareas cotidianas y retos nuevos. Según Aíra, los lentes inteligentes pueden ayudar a los usuarios a transitar por calles congestionadas, utilizar el transporte público, comprar en tiendas, o incluso escalar una montaña (SEGURA MEDRANDA, 2019).

De acuerdo con la información dada por el CONADIS y la ASOCIACION DE PERSONAS CON DISCAPACIDAD FISICA DE CHIMBORAZO precedida por el Sr. Luis Palacios las personas con discapacidad visual tienen problemas al cruzar las calles por falta de instalación de semáforos de tres tiempos, impiden a las personas ciegas saber cuándo deben cruzar una calle, la falta de señalización en el ciclo vías, veredas y calzadas en mal estado y entre otras necesidades.

Dado a nivel del país no se han desarrollado este tipo de prototipos se desea realizar el tema propuesto que es el diseño y construcción de un sistema electrónico para personas no videntes como ayuda para el cruce de las calles urbanas basado en el procesamiento de imágenes.

FORMULACIÓN DEL PROBLEMA

¿Como diseñar e implementar un prototipo embebido de reconocimiento de objetos por medio de visión artificial como ayuda a personas con deficiencia visual?

SISTEMATIZACIÓN DEL PROBLEMA

- ¿Qué tecnologías y algoritmos se aplican para visión artificial utilizando tarjetas de desarrollo?
- ¿Cuáles son los requerimientos que debe cumplir el prototipo a implementar?
- ¿Cuál es el diseño que cumple con los requerimientos establecidos?
- ¿Qué software y hardware permiten implementar el diseño del prototipo electrónico propuesto mediante visión artificial utilizando tarjetas de desarrollo?
- ¿El prototipo electrónico implementado cumple con los requerimientos establecidos?

JUSTIFICACIÓN TEÓRICA

Las estadísticas de discapacidades en Ecuador se revelan por provincias, donde la mayoría de las personas con discapacidades se encuentran en las provincias del Guayas con un aproximado de 97.703 individuos, después seguido de la provincia de Pichincha con un aproximado de 62.494 individuos. Dado que el trabajo de titulación está enfocado a las personas con deficiencia visual a nivel de Ecuador, donde detalla 13.911 individuos con algún tipo de discapacidad es viable realizar este tipo de trabajos de titulación (CONADIS, 2016b).

Los tipos de discapacidades en el Ecuador de mayor grado inician con la discapacidad auditiva con 52.508 individuos, física con 195.046 individuos, intelectual con 92.121 individuos, de lenguaje con 5.615 individuos, psicológica con 7.560 individuos, psicosocial con 10.355 individuos y visual con 58.516 individuos, evidenciando la discapacidad física como la de mayor grado (CONADIS, 2021a).

A nivel de la provincia de Chimborazo las discapacidades de mayor grado inician con la discapacidad auditiva con 1.279 individuos, física con 2.909 individuos, intelectual con 1.245 individuos, lenguaje 63 individuos, psicológica con 39 individuos, psicosocial con 135 individuos y visual con 717 individuos (CONADIS, 2021a). Se toma como principal campo las discapacidades visuales ya que en ellas se encuentra las discapacidades motrices superiores, que son a las que va dirigido el trabajo de titulación.

Estas personas encuentran una serie dificultades debido a que la mayoría de los ambientes no son aptos para ellas. Entonces, surge la necesidad de dar solución a esta problemática mediante el

desarrollo de aplicaciones basadas en metodologías y técnicas como la visión artificial que en los últimos años ha servido para la creación de sistemas de apoyo y soporte a personas con deficiencias visuales.

La visión artificial como substituto de la visión humana es una herramienta importante en el desarrollo de dispositivos de apoyo a personas ciegas y débiles visuales. Entre las tareas para las cuales se ha usado la visión artificial para apoyo a personas invidentes con resultados prometedores en los que se incluye: movilidad, orientación, reconocimiento de objetos, acceso a acceso a información impresa e interacción social (SEGURA MEDRANDA, 2019 pág. 22).

En este proyecto de investigación, se propone el diseñar y construir un sistema electrónico para personas no videntes como ayuda para movilizarse en un ambiente indo por medio de un parlante el cual nos ayuda a dar instrucciones, de esta forma ayudar a la movilización del individuo, que este basado en el procesamiento de imágenes y que sea de bajo costo.

JUSTIFICACIÓN APLICATIVA

La implementación de este prototipo aportará de manera especial a las personas invidentes para movilizarse dentro de espacios indo por medio de altavoces que guiaran a la persona tanto con instrucciones de izquierda y derecha , ya que reconoce el objeto que se encuentre cerca. El prototipo ayudara a estas personas mediante mensajes auditivos es decir este prototipo, tendrá el siguiente proceso: mediante una cámara que ejecutara capturas de imágenes, las cuales serán analizadas con un algoritmo el cual reconoce el objeto que se encuentre cerca del individuo y por medio de un altavoz dará instrucciones de izquierda y derecha para que pueda movilizarse dentro de un espacio reducido. A nivel del país se han desarrollado pocos de estos prototipos por ello es viable realizar este tipo de proyectos.

OBJETIVOS

OBJETIVO GENERAL

Diseñar e implementar un prototipo embebido de reconocimiento de objetos por medio de visión artificial como ayuda a personas con deficiencia visual.

OBJETIVOS ESPECÍFICOS

- Estudiar las tecnologías y los algoritmos existentes para visión artificial en el reconocimiento de imágenes utilizando tarjeta de desarrollo.
- Analizar y especificar los requerimientos técnicos que debe cumplir el prototipo a implementar.

- Diseñar y construir el prototipo que cumpla con los requerimientos establecidos
- Seleccionar el software y hardware que permita implementar el diseño propuesto.
- Evaluar si el prototipo electrónico implementado cumple con los requerimientos establecidos antes planteados.

CAPÍTULO I

1. MARCO TEÓRICO REFERENCIAL

En el presente capítulo se planea abarcar temáticas acorde al proyecto para entender las ideas básicas y generalidades de los elementos a tener en cuenta en el desarrollo del mismo, en el cual se definirán conceptos de importancia y definiciones, así como la discapacidad visual, orientación y movilidad para personas invidentes, las técnicas que estas personas usan para desplazase, además los elementos que se usaran para el desarrollo de este prototipo como una tarjeta de desarrollo, los parlantes , los objetos cercanos a identificar mediante un algoritmo, , entre otros conceptos que son necesarios para una mejor comprensión.

1.1 Discapacidad visual

Es la condición o falta de capacidad de ejecutar una actividad o una acción en la misma circunstancia, sé considera normal para una persona. Con arreglo de la Clasificación Internacional de Enfermedades, la función visual se subdivide en cuatro niveles (MATI, 2020):

- La discapacidad visual moderada.
- La discapacidad visual grave se reagrupa comúnmente bajo el término “baja visión”.
- La baja visión.
- La ceguera representa al mismo tiempo el total de casos de discapacidad visual.

1.1.1 Clasificación de la Discapacidad Visual

Existen varias formas de clasificar a la discapacidad visual, sin embrago con arreglo a la Clasificación Internacional de Enfermedades CIE-10 la función visual se subdivide en (MATI, 2020):

- Visión normal.
- Discapacidad visual moderada.
- Discapacidad visual grave.
- Ceguera.

1.1.1.1 Visión normal

Puede realizar tareas sin ayudas especiales, es decir que puede ver los objetos claramente sin necesidad de algún tipo de ayuda.

1.1.1.2 Discapacidad visual moderna

Puede realizar tareas casi normales con ayudas especiales, no puede visualizar los objetos de manera normal por eso necesita algún tipo de ayuda como lentes.

1.1.1.3 Discapacidad visual grave

Tiene dificultad para tareas visuales gruesas, no puede realizar tareas que exijan control de detalles.

1.1.1.4 Ceguera

Es la ausencia de luz que no se puede corregir con gafas o lentes de contacto. Las principales causas de ceguera crónica son las cataratas, el glaucoma, la degeneración macular relacionada con la edad, las opacidades corneales, la retinopatía diabética, el tracoma y las afecciones oculares infantiles, como las causadas por la carencia de vitamina A (ORGANIZACIÓN MUNDIAL DE LA SALUD, 2020 pág. 3).

1.2 Orientación y movilidad

De acuerdo con el Pronadis la "*Orientación y Movilidad es el área donde la persona con discapacidad visual adquiere las herramientas para su desplazamiento independiente. Es impartida por docentes especializados (en niños y niñas) o por instructores (en jóvenes y adultos). Es mucho más que el uso de un bastón y permite que la persona se desplace en forma autónoma, segura y elegante*" (MINISTERIO DEL DESARROLLO SOCIAL, 2013).

1.2.1 Orientación

La orientación implica la comprensión del ambiente, es decir, la toma de conciencia de la situación de la persona en el espacio y la capacidad para relacionarse espacialmente entre los elementos del ambiente (DUARTE DUARTE, 2018).

1.2.2 Movilidad

La movilidad, o capacidad para desplazarse con independencia, seguridad y eficacia. Implica el aprendizaje de técnicas de protección (para interiores y exteriores) y de otras técnicas que permiten a la persona con discapacidad visual caminar en línea recta, seguir referencias, cruzar calles, y utilizar el transporte público.

1.3 Herramientas de navegación para invidentes

Hay una variedad de instrumentos para personas invidentes que les ayuda a desplazarse por el medio o entorno que los rodea, de una manera segura y práctica entre las cuales se tienen las siguientes:

1.3.1 Bastón

Este mecanismo usualmente se usa para ubicar obstáculos y suministrar información a la persona, sobre el entorno que lo rodea al desplazarse en lugares cerrados o abiertos. Además, se identifica a la persona con una discapacidad visual (MURILLO, y otros, 2017).

1.3.2 Ayudas electrónicas

Son dispositivos electrónicos que ayudan u orientan a las personas con discapacidad visual a localizar e identificar las diferentes referencias y obstáculos a una persona ciega. Estos dispositivos se los conocen como ETAs, por sus siglas en inglés: Electronic Travel Aids. Estas tecnologías de apoyo tienen la finalidad de mejorar la orientación y movilidad de las personas ciegas durante su desplazamiento en los diferentes entornos que pueden ser cerrados o abiertos, además suelen ser conocidos o desconocidos. Se pueden encontrar diferentes dispositivos electrónicos usualmente están enfocados a una tarea específica (orientación, esquivar obstáculos, acceder a información del medio que los rodeo u otro, etc.) Estos sistemas de navegación generalmente están provistos de (ORGANIZACIÓN NACIONAL DE CIEGOS ESPAÑOLES, 2011):

- Detección de obstáculos y peligros.
- Información de ubicación y orientación durante la navegación.
- Rutas óptimas hacia un destino deseado.

1.3.2.1 Lazzus

Se trata de una herramienta con la que invidentes y personas con una alta discapacidad visual podrán conocer, a través de su dispositivo móvil, información sobre lo que tienen a su alrededor en cada momento.

Desarrollada para dispositivos Android, así como también para los móviles de Apple e incluso las Google Glass, Lazzus basa su tecnología en el GPS que se lleva metido en el bolsillo. El mismo que se convertirá en los ojos de los usuarios. El GPS del dispositivo móvil informa a Lazzus sobre la ubicación del usuario, mientras que la brújula del smartphone indica a la aplicación cuál es su orientación. A partir de ahí, la herramienta informa por medio de audios al

invidente sobre los puntos de interés que hay en ese preciso instante en su campo visual. Todo ello en tiempo real.

El usuario puede recibir información de su localización exacta en ese momento, así como de los puntos de interés que están a su alrededor (por ejemplo, el nombre de los establecimientos) gracias al campo de visión artificial que crea la 'App'. Si desea dirigirse a uno de ellos, la aplicación le orientará. No existe descripción de los componentes tanto del hardware como del software usado (LAZZUS, 2021).

1.3.2.2 SeeLight

La aplicación móvil funciona de tal manera que después que el usuario sea registrado en el servicio, puede completar un formulario que incluirá los datos relacionados con los pasos de peatones (dónde se colocan, si hay un semáforo, si hay sonidos, con qué frecuencia funcionan, entre otros parámetros. Cualquiera puede agregar semáforos al sistema con la API abierta.

Todos pueden usarlo: agregando los semáforos en el mapa, una vez que todas las entradas, la información será geolocalizada y cargada en un mapa interactivo que servirá como una guía para los ciegos para garantizar una mayor seguridad al caminar por la ciudad, y luego, hacer que la persona sea más independiente.

La aplicación anuncia en voz alta la dirección y la distancia al semáforo más cercano y notifica al usuario sobre el mismo en el modo de fondo. SeeLight se puede considerar como una aplicación social, ya que funciona con la ayuda de personas que realmente pueden ver e informar la presencia de un cruce, proporcionando esta información a las personas con discapacidad visual. No existe descripción de los componentes del hardware como del software utilizado, en la Figura 1-1 se puede apreciar la imagen de la aplicación móvil (SEELIGHT, 2020).

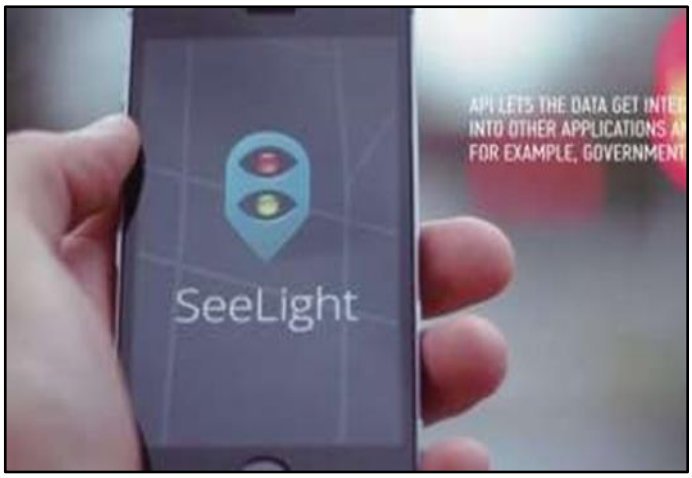


Figura 1-1: Aplicación móvil del SeeLight

Fuente: (SEELIGHT, 2020)

1.3.2.3 Dispositivo de ayuda para los ciegos UCSU

El grupo de investigación de la universidad de Carlos III de Madrid (UC3M) ha diseñado un sistema de ayuda para los ciegos, la propuesta consiste en un procesador de estereovisión que, midiendo la diferencia de imágenes captadas por dos cámaras ligeramente desplazadas, calcula la distancia a cada punto de la escena.

Después, para transmitir esa información al usuario se emplea un código de sonidos que informa de la posición y distancia de los distintos obstáculos. La cantidad de información recibida de la escena depende del perfil cognitivo seleccionado. Se puede elegir entre "seis perfiles, desde uno muy sencillo, con alarma sonora sólo cuando te vas a chocar, a otros que describen la escena con sesenta y cuatro sonidos simultáneos", indica Revuelta. El sistema aún no está en una fase comercialización, porque de momento tan solo se ha construido un prototipo en la Figura 2-1 se muestra el prototipo (REVUELTA SANZ, 2013).

Figura 2-1: Aplicación móvil

Fuente: (XATAKANDROID, 2020)

1.3.2.4 Sistema de retroalimentación para ciegos con discapacidad visual HALO

La empresa Polymythic, se dedica a la tecnología accesible que tuvo la idea de desarrollar un dispositivo de retroalimentación háptica para las personas con discapacidad visual.

Utiliza una serie de telémetros que toman datos de sensores y de la retroalimentación de salida provenientes de impulsos motores de vibración (sensores) que se encuentran colocados en una superficie de la cabeza de una persona. Cuando la persona se acerca a un objeto, la intensidad y la frecuencia de la vibración aumenta, siendo directamente proporcional a la distancia de un objeto. Si los sensores carecen de retroalimentación, entonces es seguro continuar en esa dirección.

El HALO está compuesto por un controlador Arduino Mega 2560 que funciona con una sola batería de 9V. Al ser un dispositivo muy útil, también tiene la ventaja de ser relativamente

económico. Dado que se encuentra dentro del rango asequible, no solo permanecerá como un concepto innovador, sino que seguramente se configurará como un producto maravilloso y una bendición para todos los necesitados, en la Figura 3-1 se puede visualizar el prototipo HALO (KHAN, 2010).

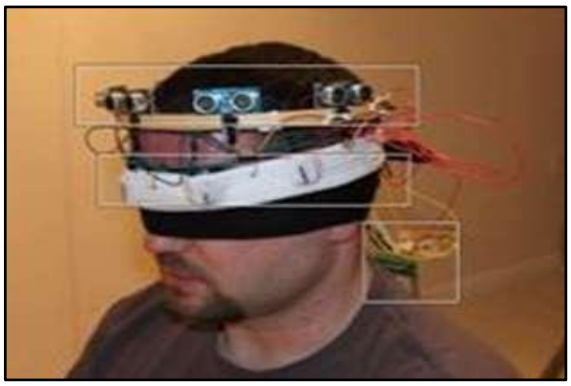


Figura 3-1: Usuario utilizando dispositivo HALO

Fuente: (ASKIX, 2020)

1.3.2.5 Dispositivo de navegación para invidentes basado en la tecnología time of flight

En el siguiente resumen se hablará de un nuevo dispositivo de navegación y detección de obstáculos para las personas ciegas, basado en la tecnología Time-of-Flight y en sonidos acústicos. El dispositivo se ha desarrollado como un dispositivo de ayuda, complementario al bastón, para las personas invidentes.

Su objetivo primordial es detectar los obstáculos e informar al usuario mediante sonidos acústicos de la locación de estos, tanto en la distancia como en dirección es decir la distancia que existe entre el usuario y el objeto además en qué dirección está el objeto. El dispositivo tiene un rango de trabajo entre los 0.5 m y 5 m en distancia y entre 30º izquierda y 30º derecha en azimut, con una precisión de 0, 9º.

El dispositivo informa a los usuarios mediante auriculares estéreo, de la presencia de los obstáculos situados en su camino. Está compuesto por un sistema de sensores 3D-CMOS montado en un par de gafas de sol, un FPGA que se encarga de procesar la información recibida por los sensores y transformarla en sonidos acústicos, y un par de auriculares estéreo.

Las pruebas experimentales llevadas a cabo demuestran el potencial que puede tener para las personas invidentes. Se prueba que con la ayuda del dispositivo acústico los usuarios se sienten más seguros a la hora de transitar por las calles, debido a que tienen una amplia información sobre el entorno que los rodea, más allá de la obtenida con el bastón. En la Figura 4-1 se puede ver como se está usando el prototipo (DUNAI, 2013).

Figura 4-1: Uso del dispositivo

Fuente: (ASKIX, 2020)

1.4 Sistemas embebidos

Un sistema embebido se lo define como cualquier equipo computacional programable, que cumple con una o varias funciones específicas en tiempo real.

"Un sistema embebido es un sistema cuya función principal no es computacional, pero es controlado por un computador integrado. Este computador puede ser un microcontrolador o un microprocesador. La palabra embebido implica que se encuentra dentro del sistema general, oculto a la vista, y forma parte de un todo de mayores dimensiones" (PÉREZ, 2009 pág. 4).

Todo tipo de sistema computacional está combinado por dos partes que son el hardware y el software embebido como los elementos más importantes. El sistema es independiente puesto que su software está embebido, suele alojarse en memoria solo de lectura (ROM) por lo que prescinde de una memoria volátil (PÉREZ, 2009 pág. 4).

En un sistema incrustado se distinguen tres características que definen y distinguen a este de otros sistemas computacionales.

- En este sistema se ejecutan tareas específicas de forma repetitiva, a diferencia de un sistema computacional ordinario que ejecuta cierta cantidad de programas.
- Guarda ciertas limitaciones en su implementación como: diseño, tamaño, costo desempeño, consumo energético. "Los sistemas embebidos generalmente deben ser poco costosos, poseer un tamaño reducido, tener un buen desempeño para procesar datos en tiempo real, y además consumir un mínimo de energía para extender el tiempo de vida de las baterías o prevenir la necesidad de elementos adicionales de enfriamiento" (PÉREZ, 2009 pág. 5).
- Capacidad de reacción ante cambios ambientales, sin interrumpir su operación ni retrasar las funciones otorgadas.

1.4.1 Componentes

Todo sistema embebido se compone de tres partes principales que son las siguientes:

1.4.1.1 Hardware

Es la parte física que constituye en sí mismo, que en conjunto con el software realizan tareas específicas. Estos componentes se distan de los sistemas tradicionales en diversos aspectos como: consumo energético, funcionalidad, tamaño, capacidad de procesamiento, y otras características (PÉREZ, 2009 pág. 5).

El hardware que comúnmente compone un sistema incrustado se describe a continuación:

- **Microprocesador / Microcontrolador:** Componente LSI (Large Scale Integration) que brinda la capacidad de cómputo al sistema y se encarga de realizar tareas o funciones en una sola pieza de circuito integrado, el mismo que se forma por miles de transistores y otros componentes electrónicos.
- **Sensores:** Dispositivo que se encarga de percibir las señales físicas de un fenómeno para traducirla a otro tipo de señales generalmente eléctricas.
- **Memoria:** Espacio de almacenamiento para la retención de datos y su posterior uso.

1.4.1.2 Software

Se define al software de un sistema embebido como la aplicación que ejecuta un trabajo en específico de forma repetitiva.

"El software que se ejecuta en un sistema embebido es diseñado bajo algunas restricciones importantes: (i) cantidades pequeñas de memoria, generalmente en el orden de los KB, (ii) capacidades limitadas de procesamiento, generalmente los procesadores poseen velocidades que no superan los Mhz, (iii) la necesidad de limitar el consumo de energía en cualquier instante, bien sea en estado de ejecución o no" (PÉREZ, 2009 pág. 6).

1.4.1.3 Sistemas operativos embebidos

Los sistemas operativos proveen de una interacción agradable entre el usuario y el sistema, y a su vez le permite al usuario administrar todos los recursos de este. Además, un sistema operativo embebido es aquel que se ejecuta bajo un sistema incrustado con características en tiempo real.

Pérez lo define como un "*software muy pequeño desarrollado específicamente para ser usado con un algún sistema embebido en particular, o en ocasiones puede ser una versión reducida de algún sistema operativo que se utiliza en una computadora de propósito general*" (PÉREZ, 2009 pág. 6).

1.5 Visión artificial

Se puede determinar la a visión artificial como, la capacidad de transmitir al ser humano la capacidad de mejorar la calidad de interpretación, en el procesamiento de la información para los sistemas o máquinas autónomas. En la figura 5-1 se identifica cada una de las etapas dentro de un proceso de visión artificial.

"*La visión artificial puede ser definida como los procesos de obtención, caracterización e interpretación de información de imágenes tomadas de un mundo tridimensional*" (PAJARES, y otros, 2001 pág. 11).

Como un concepto, Pajares y De la Cruz lo delimitan como "*la capacidad de la máquina para ver el mundo que le rodea, más precisamente para deducir la estructura y las propiedades del mundo tridimensional a partir de una o más imágenes bidimensionales"* (PAJARES, y otros, 2001 pág. 20).

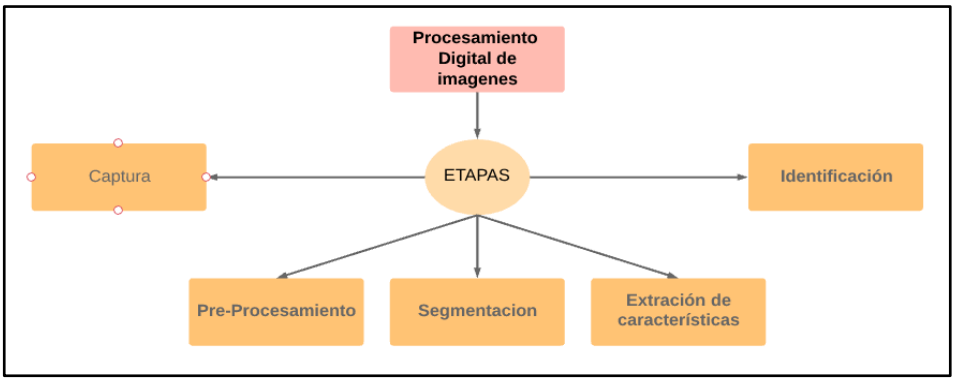


Figura 5-1: Proceso de un sistema de visión artificial

Realizado por: Guillen M., Bonilla B., 2021.

1.5.1 Arquitectura de un sistema de visión artificial

Un sistema de visión artificial está conformado de un mínimo de cinco elementos:

- **Dispositivo de captura:** Unidad óptica que recepta las señales analógicas del mundo real.
- **Conversión A/D:** La señal analógica obtenida se convierte a una o varias señales digitales.
- **Memoria:** Donde se almacena la información obtenida.
- **Procesador:** Trabaja en conjunto con la memoria para operar sobre la imagen digitalizada.
- **Monitor:** Permite la visualización de los procesos realizados, aunque se puede prescindir del mismo.

1.5.1.1 Dispositivos de captura de imágenes

Para adquirir una imagen a partir de una escena, es fundamental un dispositivo físico que sea sensible a cierta banda del espectro electromagnético, que produzca una señal eléctrica en proporción a la energía percibida por el sensor, para luego ser digitalizada, en la Figura 6-1 se describe el proceso espectro electromagnético **(PAJARES, y otros, 2001 pág. 23)**.

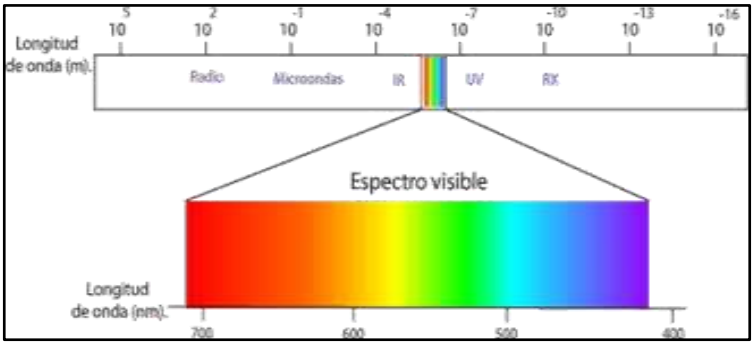


Figura 6-1: Espectro electromagnético

Fuente: (AULACLIC, 2020)

Entre los principales dispositivos de captura se tienen: cámaras fotográficas, scanner, sensores de rango y ultrasonido, rayos X, imágenes de tomografías y resonancias, etc.

1.5.1.2 Digitalización de imágenes

Es el proceso por el cual se convierte las señales analógicas obtenidas por el sensor a una señal continua, es decir a partir del muestreo de la imagen obtenida se convierte a una matriz discreta de MxN píxeles.

1.5.2 Aplicaciones de la visión artificial

En la actualidad existe una gran variedad de aplicaciones en cuanto a visión artificial se refiere, como se puede apreciar en la tabla 1- 1.

Tabla 1-1: Aplicaciones de la visión artificial

Inspección industrial y control de calidad	Verificación de etiquetado y códigos. Inspección de soldaduras, circuitos impresos, motores.
Vigilancia y seguridad	Control de accesos. Control de abandonos.
Identificación	Identificación biométrica: huellas, pisadas, firmas, iris. Reconocimiento de caras, de gestos, caracteres.
Control de tráfico	Reconocimiento de matrículas, peaje por volumen, control de flujo. Sistemas de ayuda a la conducción.
Guiado de robots	Industriales Vehículos autónomos
Análisis de imágenes	Satélite
Aplicaciones militares	Detección de objetivos

Fuente: (SALGADO, 2007)

Realizado por: Guillen M., Bonilla B., 2021.

1.5.3 Representación de una imagen

La imagen obtenida por cualquier dispositivo de captura se presenta digitalizada en forma de una matriz de M x N elementos; representado a través de un sistema coordenado f(x,y); mostrando su origen en el extremo superior izquierdo, como se puede apreciar en la Figura 7-1. A cada elemento de este arreglo se lo denomina pixel (picture element) (PAJARES, y otros, 2001 pág. 30).

Todos los píxeles guardan cierta información, es decir en el caso de que la imagen esté en niveles de grises dicha información será el brillo.

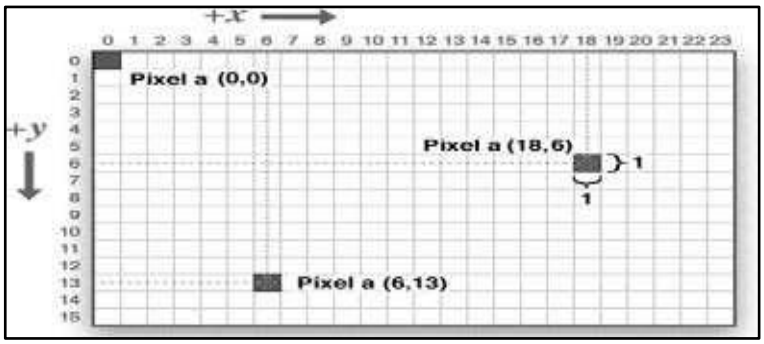


Figura 7-1: Sistema de coordenado de una imagen

Fuente: (EPA, 2018)

Las coordenadas de un píxel (y) tiene dos vecinos horizontales y verticales, a este conjunto se le denomina, vecindad a cuatro de un píxel, en la Figura 8-1 se puede apreciar la vecindad de un pixel.

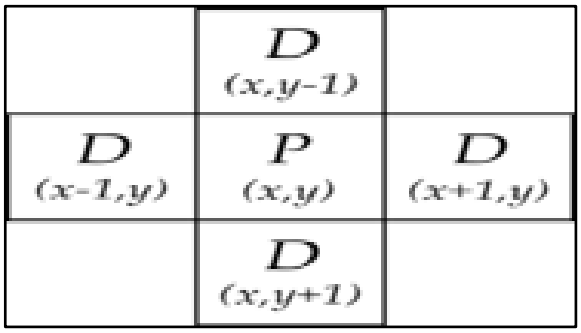


Figura 8-1: Vecindad a cuatro de un pixel

Fuente: (EPA, 2018)

Cada uno de ellos tiene una distancia unitaria desde su posición original. Las coordenadas vienen dadas por la Ecuación 1-1.

$$(x+1,y),(x-1,y)(x,y+a),(x,y-1) \qquad \textbf{Ecuación 1-1}$$

1.5.4 Segmentación

La imagen que provee un dispositivo de captura habitualmente no es lo suficientemente buena para extraer información de manera adecuada, por lo que se tiene que apartar los fragmentos o características de la imagen para facilitar su análisis o reconocimiento.

"La segmentación es el proceso por el cual a través de técnicas y procedimientos se extrae información útil de una imagen, para su posterior uso. Este proceso se basa en la similitud (se orienta hacia regiones) y discontinuidad (se orienta hacia bordes)" (PAJARES, y otros, 2001 pág. 8).

1.5.4.1 Binarización

Es el proceso por el cual se compara los niveles de grises de una imagen con un valor determinado conocido como umbral (threshold). Si el nivel de gris presente en el píxel es menor al umbral, a este se le asigna un valor de cero lo que supone el color negro Ecuación 2-1, de manera análoga, si el píxel es mayor al umbral esta toma el valor de uno o blanco Ecuación 3-1.

$$S[x,y] = 0(0), E\ [x,y] \leq T \qquad \textbf{Ecuación 2-1}$$

$$S[x,y] = 1(255), E\ [x,y] > Tm \qquad \textbf{Ecuación 3-1}$$

El problema principal de la binarización es el de hallar el valor de umbral adecuado para que se tome valores de cero o uno, ya que el nivel de gris de una imagen puede variar por diversos factores y eliminar información que podría ser útil para el sistema, lo cual podría provocar ruido al momento de hacer un análisis, en la Figura 9-1 se puede apreciar la binarización de una imagen y además la inversión de los colores.

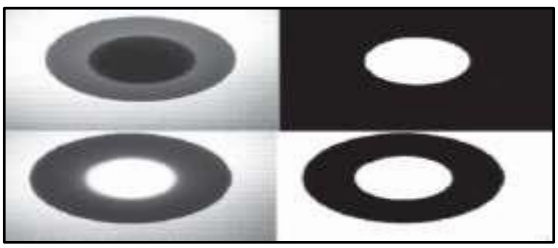


Figura 9-1: Binarización

Fuente: (EPA, 2018)

1.5.4.2 Detección de contornos

La detección de contornos es una técnica aplicada al procesamiento de imágenes de donde se extrae información de los límites de una forma o un objeto, para ser usados después para la

detección y análisis de este. Se basa en las variaciones de la intensidad de un pixel, si éste pasa de un valor de 255 (blanco) a 0 (negro) o viceversa, representa un contorno.

De acuerdo con, Vélez mencionan que: *"El contorno de un objeto en una imagen digital corresponde al mínimo conjunto de píxeles que separa ese objeto del fondo o background de la imagen. Normalmente estos contornos se corresponden con los puntos donde se producen discontinuidades en los valores de píxeles adyacentes (cambios en el matiz o el brillo) o con los puntos donde cambia un patrón que se repite (cambios de textura)*" (VÉLEZ, 2007 pág. 128).

1.5.5 Operaciones morfológicas

Dentro del campo de del procesamiento de imágenes, la morfología se entiende como de la topología o la estructura de objetos a partir de su respectiva imagen. Las operaciones morfológicas se encargan del realce de la geometría de los pixeles especialmente en la teoría de conjuntos.

1.5.5.1 Erosión

La erosión es la transformación morfológica que combina dos conjuntos, usando el vector resta de los elementos del conjunto. Esta operación degrada la imagen fuente, eliminando los píxeles vecinos que toman valor de uno en la imagen fuente, como se aprecia en la Figura 10-1. De acuerdo con Vélez define la erosión como "*el conjunto de todos los elementos x para los cuales B trasladado por x está contenido en A*" (VÉLEZ, 2007 pág. 130).

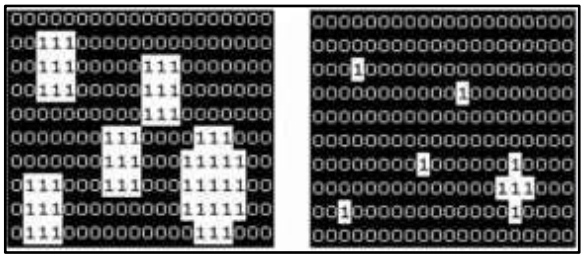


Figura 10-1: Erosión de una imagen binaria

Fuente: (EPA, 2018)

Matemáticamente la erosión se define como:

$$A\theta B = \{x/Bx \in A) \qquad \textbf{Ecuación 4-1}$$

1.5.5.2 Dilatación

La dilatación es la trasformación morfológica que combina dos conjuntos usando adición de vectores lo contrario de la erosión, de los elementos del conjunto. La dilatación representa una

ampliación de la imagen fuente, donde todos los píxeles de la vecindad toman un valor de uno, como se aprecia en la Figura 11-1.

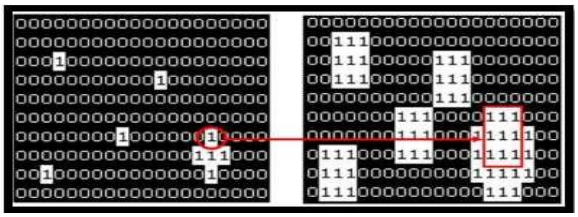


Figura 11-1: Dilatación de una imagen binaria

Fuente: (EPA, 2018)

Esta operación da como resultado a un conjunto de elementos, donde al menos un elemento de B es contenido en la Ecuación 5-1. Matemáticamente la dilatación se define como:

$$A\theta B = \{x/(\mathrm{BA})_{x} \cap \mathrm{A}\} \qquad \textbf{Ecuación 5-1}$$

1.6 Software para el desarrollo del sistema

El lenguaje Python posee características que permiten crear diferentes tipos de programas, se descompone en sentencias, expresiones, objetos y módulos. Esto se muestra en la figura 12-1

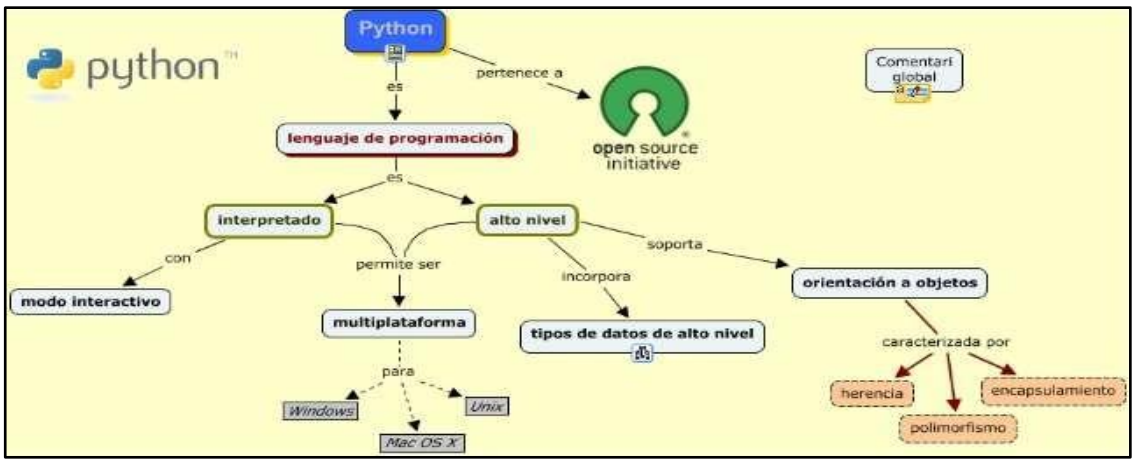


Figura 12-1: Características de Python

Fuente: (OPENWEBINAR, 2019)

Tiene características como:

- **Interpretado:** No compila el código antes de ser ejecutado.
- **Modo interactivo:** La sentencia se ejecuta y el resultado se visualiza, esto facilita al programador a verificar el código.
- **Multiplataforma:** Desarrollado para sistemas informáticos.
- **Orientado a objetos:** Crea programas con códigos reutilizables.

1.6.1 OpenCv

La librería OpenCv (Open Source Computer Vision) se emplea para el procesamiento de imágenes en visión artificial, se emplea también en sistemas de seguridad, programas de detección de movimiento, reconocimiento de objetos, control de procesos, robótica entre otros. Se considera una librería gratuita y libre escrita en C y C++, se emplea con cualquier tipo de procesador y funcionan con sistemas como Windows, Linux, Android, Mac OSX e iOS, por lo cual permite crear aplicaciones en tiempo real (MARÍN, 2020).

Tabla 2-1: Componentes principales de OpenCv

CV	Contiene algoritmos que permite el procesamiento de imágenes y visión porordenador
Ml	Posee herramientas de agrupación y clasificadores estadísticos.
HighGUI	Realiza funciones de entrada y salida (I/O) que permite subir y almacenarimágenes o videos.
CXCore	Contiene algoritmos con funciones gráficas. Permite recibir información de CV, MIL Y HighGUI.
CvAux	Posee algoritmos experimentales.

Fuente: (SALGADO, 2007)

Realizado por: Guillen M., Bonilla B., 2021

La biblioteca OpenCV utiliza funciones que permite identificar personas u objetos, inspección de objetos, reconocimiento de rostro, reconstrucción 3D entre otros, esto debido a la infraestructura accesible sobre el procesamiento de imágenes.

1.6.2 SIFT & FLANN

Los algoritmos SIFT & FLANN se pueden realizar en Matlab, Python y C.

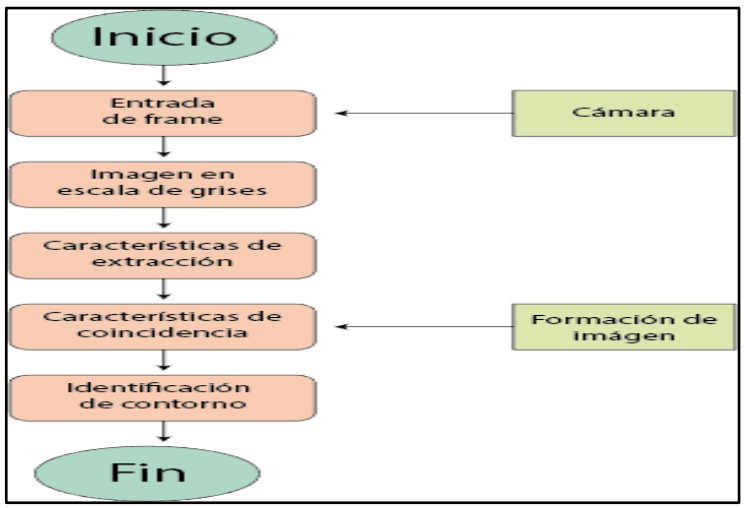


Figura 13-1: Algoritmo SIFT & FLANN para detección de objetos

Realizado por: Guillen M., Bonilla T., 2021.

La función básica de FLANN es realizar búsquedas cercanas de pixeles en espacios de gran dimensión esto permite crear sistemas de capturas de imágenes por contornos en distintos campos. El algoritmo SIFT desarrolla el método de coincidencia de imágenes por detección de esquinas, existen distintas aplicaciones para este algoritmo entre una de ellas es el reconocimiento facial. Un ejemplo se indica en la figura 14-1.

Los descriptores del algoritmo SIFT busca las coordenadas en puntos establecidos para direccionarlos a una orientación determinada, cada descriptor debe ser diferente entre ellos. Poseen histogramas que calculan valores de magnitud en dimensiones de 16 x 16 pixeles de 8 bits por cada uno (PROGRAMADOR CLIC, 2018).

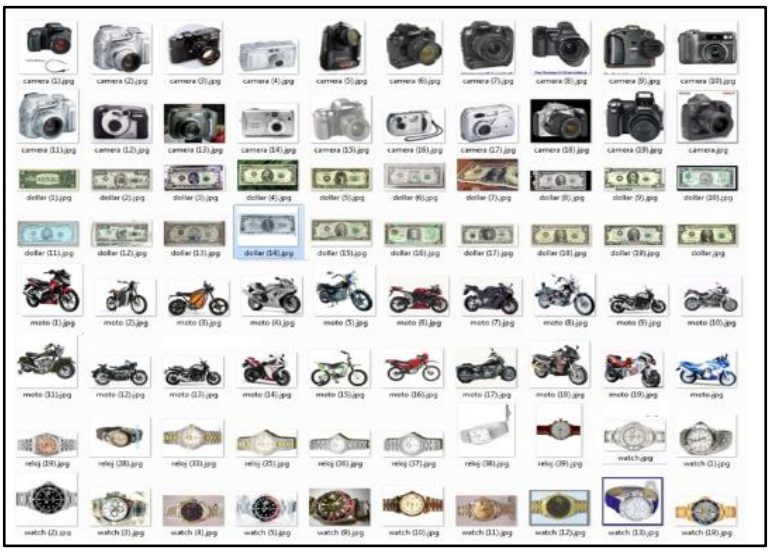


Figura 14-1: Clasificación de objetos mediante SIFT

Realizado por: Guillen M., Bonilla T., 2021.

El algoritmo SIFT permite agrupar objetos de forma aglomerada, cada agrupamiento es iniciado por un descriptor el cual selecciona a todos los objetos de un mismo tipo en un solo grupo. La similitud entre objetos se vincula de manera simple, media o completa en el cual se obtiene valores de umbral para detener el proceso.

1.6.3 HARR Cascade

La clasificación de imágenes por medio del algoritmo HAAR Cascade permite la detección de objetos bajo términos de escala y forma, en la etapa de entrenamiento procede a clasificar objetos en diferentes locaciones mediante parámetros de rechazo estadístico (LUDEÑA, 2019). Esto se muestra en la figura 15-1.

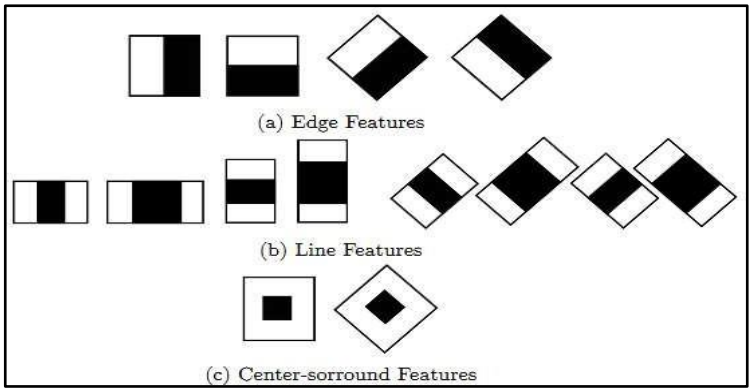


Figura 15-1: Clasificador de patrones mediante HAAR Cascade

Fuente: (LUDEÑA, 2019)

Los principales clasificadores son creados a partir de patrones pequeños los cuales se van acumulando para aumentar muestras clasificadas en las próximas interacciones. Estas acumulaciones crean patrones que calculan sectores que poseen más iluminación y restan sectores oscuros eliminando probabilidades de muestras falsas.

La información anteriormente recopilada se toma como punto de partida para la creación de un sistema de supervisión para niños, empleando los software y hardware necesarios que permitan el desarrollo de este.

1.7 Hardware para el desarrollo del sistema embebido

Como se requiere reducir costos para el prototipo a construir, se buscan tarjetas de desarrollo de bajo costo, se busca un hardware que permita el desarrollo del sistema a implementar en el trabajo de titulación, usando todos los recursos necesarios para el mismo.

1.7.1 Raspberry Pi

En la Figura 16-1, se puede visualizar el ordenador al cual se lo puede determinar como un ordenador de tamaño reducido de crédito de bajo costo, diseñado por la fundación Raspberry Pi con el fin de ser usado para muchas cosas como la enseñanza, programación, hoja de cálculo etc. Se conecta a su televisor y un teclado. Es una placa que soporta varios componentes necesarios en un ordenador común (RASPBERRY PI, 2018a).

Figura 16-1: Raspberry Pi

Fuente: (RASPBERRY PI, 2018a)

1.7.1.1 Características principales de las tarjetas de desarrollo

Existe una variedad de estas tarjetas de desarrollo que se las detalla en la siguiente tabla 3-1.

Tabla 3-1: Características diferentes Raspberry

Características	Raspberry Pi 4 Modelo B	Raspberry Pi 3 Modelo B+	Raspberry Pi 3 Modelo B
Procesador	Broadcom BCM2711B0, quad-core Cortex-A72	Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC	Broadcom BCM2837, Cortex-A53 (ARMv8) 64-bit SoC
Frecuencia de reloj	1,5 GHz	1,4 GHz	1,2 GHz
Gpu	VideoCore VI 500 MHz	VideoCore IV 400 MHz	VideoCore IV 400 MHz
Memoria	1/2/4 GB LPDDR4-3200	1GB LPDDR2 SDRAM	1GB LPDDR2 SDRAM
Conectividad inalámbrica	Wi-Fi 2,4GHz / 5GHz IEEE 802.11.b/g/n/ac Bluetooth 5.0, BLE	Wi-Fi 2,4GHz / 5GHz IEEE 802.11.b/g/n/ac Bluetooth 4.2, BLE	Wi-Fi 2,4GHz IEEE 802.11.b/g/n Bluetooth 4.1
Conectividad de red	Gigabit Ethernet	Gigabit Ethernet over USB 2.0 (300 Mbps de máximo teórico)	Fast Ethernet 10/100 Gbps
Puertos	GPIO 40 pines 2 x Micro HDMI 2 x USB 2.0 2 x USB 3.0 CSI (cámara Raspberry Pi) DSI (pantalla tácil) Toma auriculares / vídeo compuesto Micro SD USB-C (alimentación) Power-over-Ethernet (PoE)	GPIO 40 pines HDMI 4 x USB 2.0 CSI (cámara Raspberry Pi) DSI (pantalla tácil) Toma auriculares / vídeo compuesto Micro SD Micro USB (alimentación) Power-over-Ethernet (PoE)	GPIO 40 pines HDMI 4 x USB 2.0 CSI (cámara Raspberry Pi) DSI (pantalla tácil) Toma auriculares / vídeo compuesto Micro SD Micro USB (alimentación)
Fecha de lanzamiento	24/06/2019	14/3/2018	29/2/2016

Fuente: (RASPBERRY PI, 2018a)

Realizado por: Guillen M., Bonilla B., 2021

1.7.1.2 Selección de tarjeta de desarrollo mediante una comparación de características

En la tabla 4-1 se muestra la comparación más alta entre Raspberry Pi y Banana Pi.

Tabla 4-1: Comparaciones entre Raspberry Pi y Banana Pi

Características	Raspberry Pi4	Banana Pi
Procesador	Broadcom CM2711B0, quad-core Cortex-A72	Allwinner A20 - Cortex A7 - 1 GHz – doble núcleo
RAM	4 GB LPDDR2	1 GB DDR3
USB	4 x USB 2.0	2 x USB 2.0, 2 x Micro USB
GPIO	Encabezado de expansión de40 pines	Encabezado de expansión de26 pines
Red	Gigabit Ethernet	1 x 10/100/1000 Ethernet

Fuente: (RASPBERRY PI, 2018a)

Realizado por: Guillen M., Bonilla B., 2021

En conclusión, el diseño de la Raspberry Pi está girando en torno a su presupuesto y facilidad de uso, que es la razón principal de su enorme demanda en el mercado. El soporte generalizado de otro software solo complica la utilidad del dispositivo, con mucha información sobre cómo escribir y compilar software, hardware creado para Raspberry Pi. Si bien también existe la misma información para Banana pi, pero no es tan común, lo cual es complicado ya que Banana Pi es un poco más difícil de configurar que Raspberry Pi.

1.7.2 Módulo de cámara

Esta cámara tiene un sensor Omnivision de 8 píxeles que permite realizar capturas de 2.592 x 1.944 píxeles y que también captura vídeo a 1080p con una tasa de 30 fotogramas por minuto. En la Figura 17-1 se muestra la cámara Raspicam que es compatible en un 100 % con cualquier versión o modelo de tarjeta Raspberry pi, ya que trae sus librerías ya preinstaladas en el sistema operativo de la tarjeta Raspberry pi (HARDZONE, 2018).

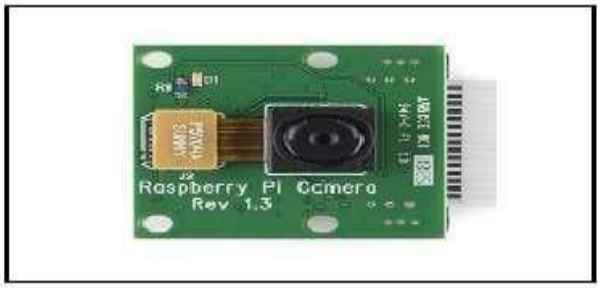


Figura 17-1: Cámara Raspicam

Fuente: (RASPBERRY PI, 2018a)

CAPÍTULO II

2. PROPUESTA Y DISEÑO DEL PROTOTIPO

En el siguiente capítulo se detalla el diseño de *hardware y software* para el prototipo a implementar, donde se da detalle de cada elemento con sus características que se usaron en el prototipo. También se detalla las etapas del proceso, así como los esquemas del sistema.

Es un sistema fundamentado en visión artificial, se considera la adquisición de una cámara la cual permita adquirir imágenes del sitio de interés.

2.1 Arquitectura general del sistema

Como se muestra en la figura 1-2, la arquitectura general del prototipo a implementar se describe de manera universal el funcionamiento de las diferentes etapas del sistema, desde el primer bloque la cual es adquirir imágenes por medio de una cámara para posteriormente pasar albloque de procesamiento de imágenes, donde se realizará el tratamiento o procesamiento de estas mediante un algoritmo para determinar las señales. Y en el último bloque se obtendrá la información mediante un mensaje de voz.

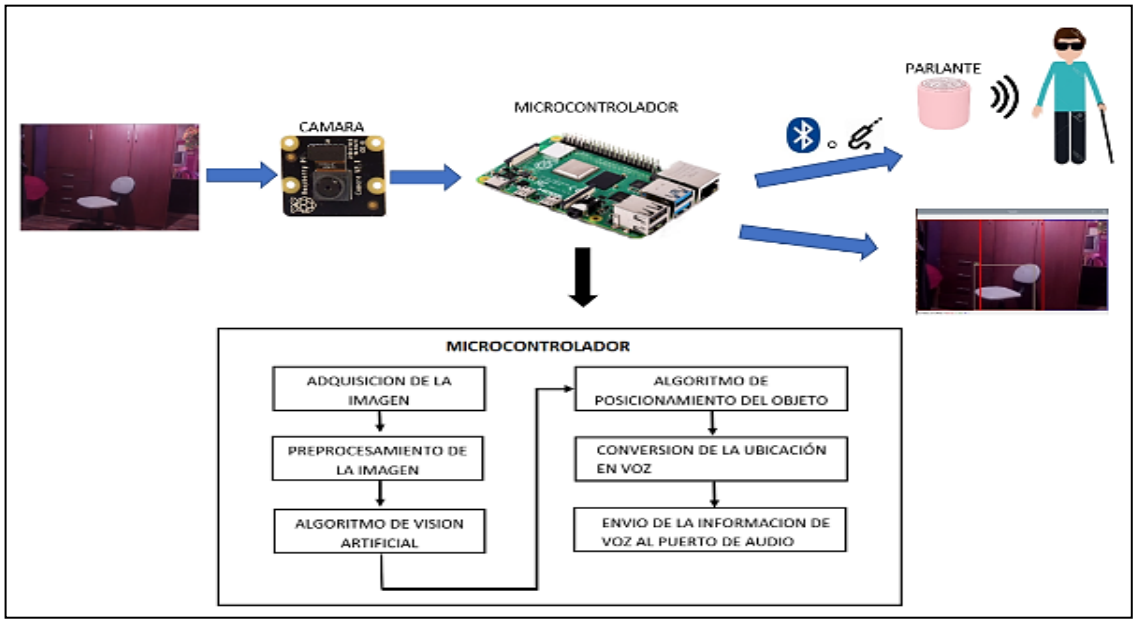


Figura 1-2: Arquitectura general del sistema

Realizado por: Guillen M., Bonilla T., 2021.

De acuerdo con la arquitectura del prototipo el cual está constituido por tres partes principales las cuales se describen a continuación cada una de ellas. En el primer bloque se tiene la adquisición imágenes mediante una cámara Pi HD Pi NoIR, en elsegundo bloque se procesa todas las imágenes adquiridas por la etapa anterior, mediante una tarjeta de desarrollo Raspberry Pi 4 y por

último en el tercer bloque se tiene él envió de mensajesde voz con la información detectada en la imagen.

2.2 Diagramas de bloques del procesamiento de las imágenes

Dentro del segundo bloque es donde se realizará el sistema de visión artificial la cual tiene varias etapas para llegar al resultado deseado cada una de las etapas son importante por eso se detallará cada una de ellas:

- La primera fase, es guardar las imágenes provenientes de la cámara como se muestra en la figura 2-2.
- La segunda fase consiste en el tratamiento digital de las imágenes, con objeto de facilitar las etapas posteriores. En la etapa de procesamiento previo es donde, mediante filtros y transformaciones geométricas, se eliminan partes indeseables de la imagen o se realzan partes interesantes de la misma, esto se muestra en la figura 2-2.
- La tercera fase junto con la cuarta se conoce como segmentación y extracción, y consiste en aislar los elementos interesados de una escena para comprenderla como se indica en la figura 3-2.
- Después, se llega a la etapa de comparación mostrada en la figura 4-2. En ella se pretende distinguir los objetos segmentados, gracias al análisis de ciertas características establecidas previamente.
- Una vez hecha la comparación se envía un mensaje el cual sería la última etapa del procesamiento de cada imagen obtenida de la primera etapa. Mostrado en la figura 5-2.

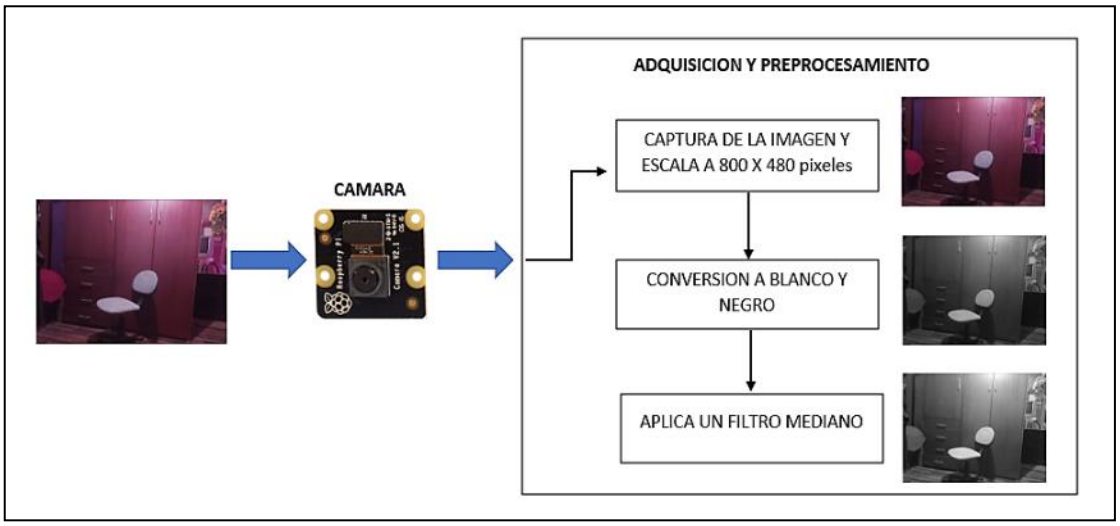


Figura 2-2: Diagrama de bloques de adquisición y procesamiento de las imágenes

Realizado por: Guillen M., Bonilla T., 2021.

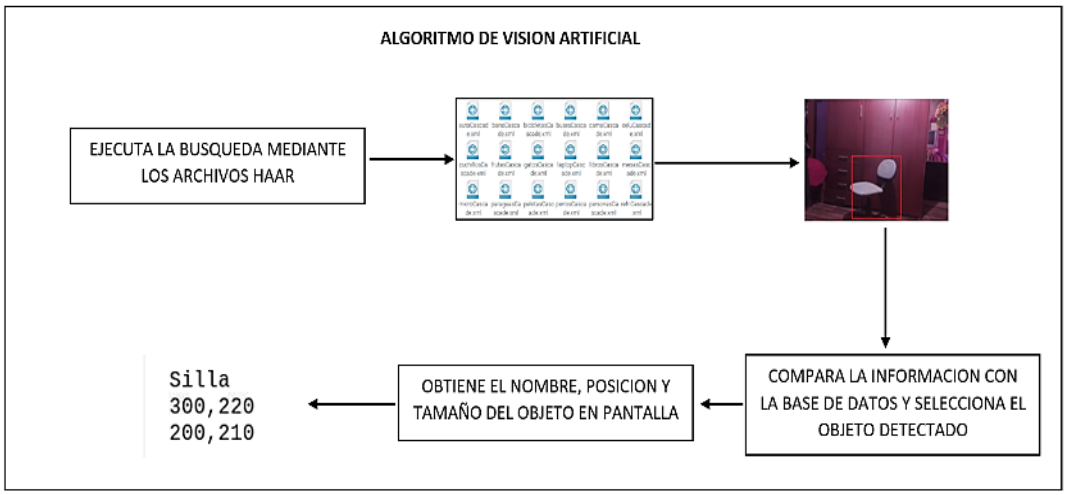


Figura 3-2: Diagrama de bloques del algoritmo de visión artificial

Realizado por: Guillen M., Bonilla T., 2021.

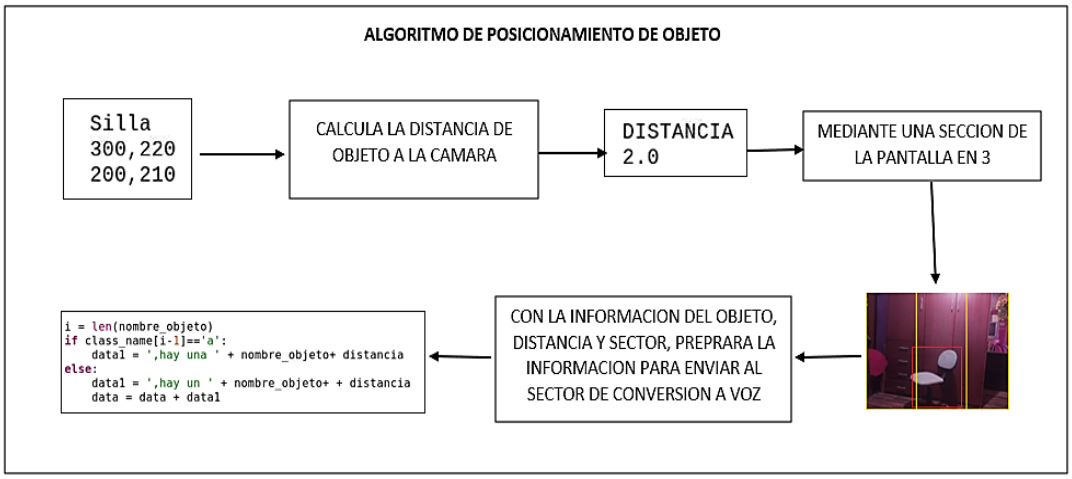


Figura 4-2: Diagrama de bloques del algoritmo de posicionamiento de objeto

Realizado por: Guillen M., Bonilla T., 2021.

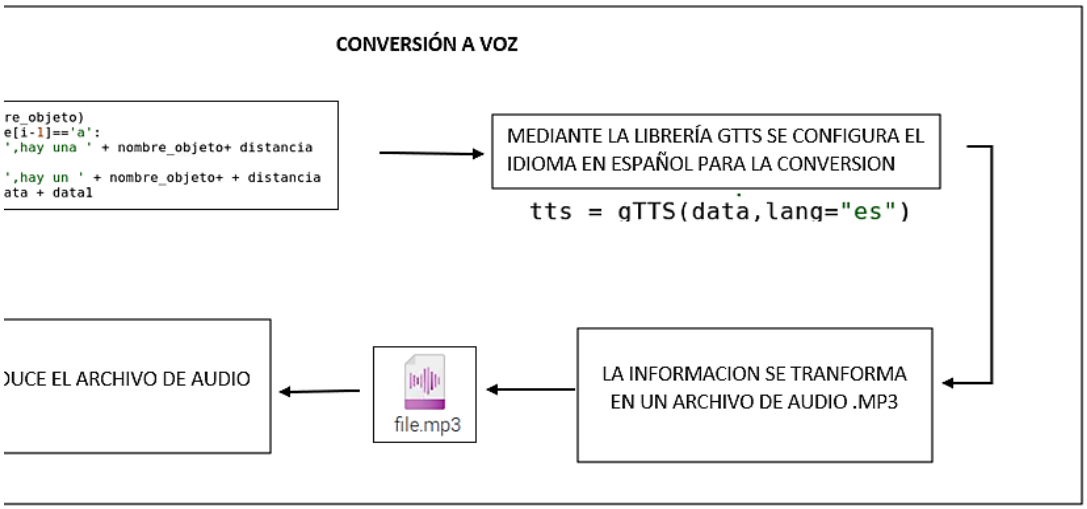


Figura 5-2: Diagrama de bloques de la conversión a voz

Realizado por: Guillen M., Bonilla T., 2021.

2.3 Requerimientos funcionales del prototipo

Para el prototipo se deben satisfacer los siguientes requerimientos funcionales:

- Detección de los objetos cercanos por medio de una cámara.
- Generar alertas auditivas una vez detectado el objeto atreves de un parlante.

2.4 Requerimientos del sistema

El prototipo debe satisfacer los requerimientos siguientes:

- Posibilitar la instalación y movilización
- Adquirir las imágenes con resolución apropiada para una detección adecuada
- Dar al sistema la capacidad necesaria para evitar posibles retardos a la hora del procesamiento.
- Admitir y procesar los datos correspondientes a la captura de la imagen.
- Establecer límites del entorno en la que se realiza la captura.
- Encontrar la ubicación del objeto para guiar al usuario a su desplazamiento
- Establecer la alerta de voz por medio del parlante para que el usuario pueda desplazarse en el entorno después de identificar los objetos que se encuentran a su alrededor.

2.5 Selección de los elementos para la construcción del prototipo

A continuación, se detallará las características de cada uno de los elementos seleccionados para el prototipo que forman parte del sistema.

2.5.1 Elementos hardware del prototipo

Dentro del mercado se puede encontrar una gran gama de dispositivos electrónicos para diversas aplicaciones, para el prototipo implementado se eligieron los siguientes elementos, por sus características, su uso, conexión, tipo de sensor y demás peculiaridades.

2.5.1.1 Cámara Pi NoIR

En la figura 6-2 se presenta un tipo de cámara para la placa de cámara Raspberry Pi HD Pi NoIR se conecta a cualquier Raspberry Pi (1,2,3,4 Zero) o Compute Module para crear fotografías y vídeo HD. La Pi NoIR (sin infrarrojos) es igual que el módulo de cámara estándar, pero sin filtro de infrarrojos. Por tanto, es excelente para fotografía y vídeo en la oscuridad o crear *Infragram.* Las características técnicas de la cámara se establecen en la tabla 1-2.

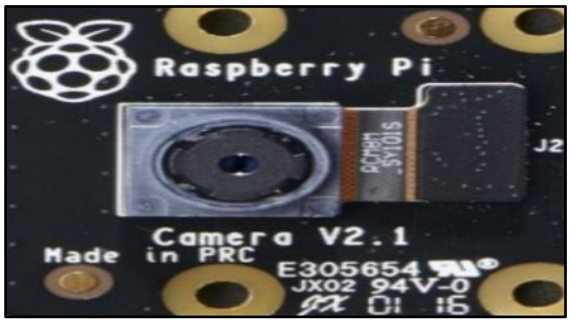


Figura 6-2: Cámara Pi NoIR

Realizado por: Guillen M., Bonilla T., 2021.

Tabla 1-2: Características técnicas principales cámara Pi NoIR

Características	Descripción
Peso	3g
Resolución	8 Megapíxeles
Modos de video	1080p30, 720p60 y 640 × 480p60 / 90
Integración Linux	Controlador V4L2 disponible
API de programación C	OpenMAX IL y otros disponibles
Sensor	Sony IMX219
Resolución del sensor	3280x2464 píxeles
Área de la imagen del sensor	3,68 x 2,76 mm (4,6 mm en diagonal)
Tamaño de Pixel	1,12 µm x 1,12 µm
Tamaño Óptico	1/4 "
Longitud Focal	3,04 milímetros
Campo de Visión horizontal	62,2 grados
Campo de Visión Vertical	48,8 grados
Relación focal (F-Stop)	2.0

Fuente: (RASPEBERRY PI, 2016b)

Realizado por: Guillen M., Bonilla T., 2021.

2.5.1.2 Tarjeta de desarrollo Raspberry Pi 4

Seleccionada como parte del prototipo porque sus características se convenientes para los requerimientos del prototipo. La tarjeta es más rápida que sus antecesores, por ese motivo fue elegida para el desarrollo del proyecto. Cuenta con un procesador de cuatro núcleos el cual es más poderoso a los primeros, se puede apreciar en la figura 7-2. Las características de la tarjeta de desarrollo Raspberry Pi 4 se establecen en la tabla 2-2.

Figura 7-2: Tarjeta de desarrollo Raspberry Pi 4

Realizado por: Guillen M., Bonilla T., 2021.

Tabla 2-2: Características técnicas principales Raspberry Pi 4

Características	Descripción
Memoria (SD RAM)	4 Gb
Unidad central de proceso	1.2 GHz, Arquitectura de 64 bits Cuatro núcleos ARMv8
Juego de instrucciones	RISC de 32 bits
Puertos USB 2.0	4
Entradas de video	Conector MIPI CSI, permite alojar el módulo de cámara
Almacenamiento	Micro SD
Conectividad a la red	Ethernet RJ45, WIFI 802.11n, Bluetooth 4.1
Consumo energético	700 mA, 3,5W
Fuente de alimentación	5V vía micro USB
Dimensiones	2,5 mm x 53,98 mm

Fuente: (RASPBERRY PI, 2018a)

Realizado por: Guillen M., Bonilla T., 2021.

2.5.1.3 Batería Li-po 7,4 V

Es una batería que tiene la capacidad de alimentar a los bloques del sistema por un tiempo determinado, para que puedan funcionar todos sus componentes de forma alámbrica, en la figura 8-2 se puede apreciar la batería Lipo a utilizar. Las características principales se indican en la tabla 3-2. Para conocer la capacidad necesaria de la batería, primero se sumó las corrientes eléctricas de los dispositivos que se alimentaran de la batería, como se muestra en la Tabla 4-2.

Figura 8-2: Batería Lipo 7,4 V

Realizado por: Guillen M., Bonilla T., 2021.

Tabla 3-2: Características técnicas principales de la batería Lipo de 7,4V

Características	Descripción
Voltaje de salida	7,4 V
Capacidad	1000 mAh
Peso	67 g
Puertos USB	72x34x15 mm

Fuente: (MGSYSTEM, 2021)

Realizado por: Guillem M., Bonilla T., 2021.

Tabla 4-2: Consumo de energía

N°	Corriente consumida Stand-By (A)	Corriente consumida con el Algoritmo (A)
1	0,15	0,52
2	0,13	0,53
3	0,15	0,6
4	0,14	0,56
5	0,15	0,57
Promedio	0,144	0,556

Realizado por: Guillem M., Bonilla T., 2021.

Con el valor total de las cargas que estarán conectadas a la batería se pude determinar que batería será la indicada para que abastezca dicho consumo y para poder determinar la durabilidad del alimentador portátil con el dispositivo conectado. Calculamos la potencia instantánea consumida por el equipo con la utilización de la ecuación 6-1.

$$P = I * V = (0{,}556\ A) * (5\ v) = 2{,}78\ W$$ **Ecuación 6-1**

Después transformamos los mAh en Wh de la batería en pocas palabras el consumo. Esto se realiza con la ecuación 7-1.

$$P = I * V = (1Ah) * (7{,}4v) = 7{,}4\ Wh$$ **Ecuación 7-1**

Entonces, en la ecuación 8-1 se indica el tiempo de autonomía que es:

$$Tiempo\ Autonomia = \frac{Consumo\_Bateria}{Potencia} = \frac{7{,}4Wh}{2.78W} = 2{,}66\ h$$ **Ecuación 8-1**

El tiempo de autonomía calculado es de 2,66 h, pero este valor puede variar ya que la batería tiene una propiedad regenerativa la cual hace que la batería se descargue con menor facilidad haciendo que el tiempo de autonomía crezca.

2.5.1.4 Memoria Micro SD

Como se muestra en la figura 9-2. La memoria Micro SD es un elemento de almacenamiento el cual tiene un papel muy importante, en la memoria se va a grabar el sistema operativo, el cual proveerá el funcionamiento a la tarjeta Raspberry pi 4, una vez instaló el sistema operativo en la memoria se inserta en la ranura de la tarjeta Raspberry pi 3 para luego iniciar con las configuraciones en la tarjeta.

Figura 9-2: Memoria Micro SD

Fuente: (GRUPOELECTROSTORE, 2019)

2.5.1.5 Parlante

Como se muestra en la figura 10-2 el parlante permite trasmitir un mensaje de voz hacia el usuario y él pueda recibir la información proveniente de la tarjeta de desarrollo Raspberry Pi 4.

Figura 10-2: Parlante

Realizado por: Guillen M., Bonilla T., 2021.

2.5.1.6 Step Down XL4015 5A

Es un dispositivo que ayuda a la carga de baterías de litio ajustando el voltaje de carga con respecto a las características de la batería, contando con un voltímetro y amperímetro permitiendo visualizar el voltaje de salida y corriente que requiere la misma, contando este con dos *Trimpot´s* que permiten ajustar el voltaje de salida y la corriente. Este dispositivo es mostrado en la figura 11-2. Las características del *Step Down* se las presenta en la tabla 5-2.

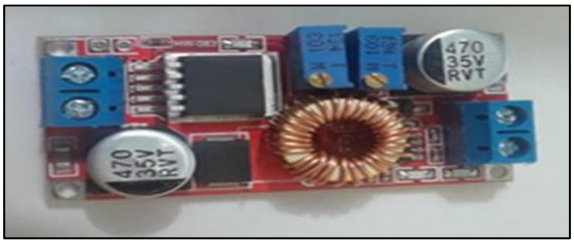


Figura 11-2: *Step Down* XL4015 5A

Realizado por: Guillen M., Bonilla T., 2021.

Tabla 5-2: Características técnicas principales del *Step Down* XL4015 5A

Características	Descripción
Voltaje de entrada	8 - 36 V(No emita más de 38 V)
Voltaje de salida	1,25 - 32 V continuamente ajustable
Corriente de salida	0 – 5 A
Potencia de salida	75 W
Temperatura de funcionamiento	-40 a +85 grados
Frecuencia de funcionamiento	180 KHz
Eficiencia de conversión	hasta 96%
Protección contra cortocircuitos	Sí (Corriente de límite 8A)
Protección contra sobre temperatura	Apaga automáticamente la salida después de sobre temperatura
Instalación	Dos tornillos de 3mm
Dimensiones del módulo	54mm (Largo) x 23mm (Ancho) x 15mm (Espesor)

Fuente: (UNIT ELECTRONICS, 2020)

Realizado por: Guillem M., Bonilla T., 2021.

2.5.2 Esquema de conexión electrónico

Este esquema en donde se indica las conexiones de los elementos esta mostrado en la figura 12-2.

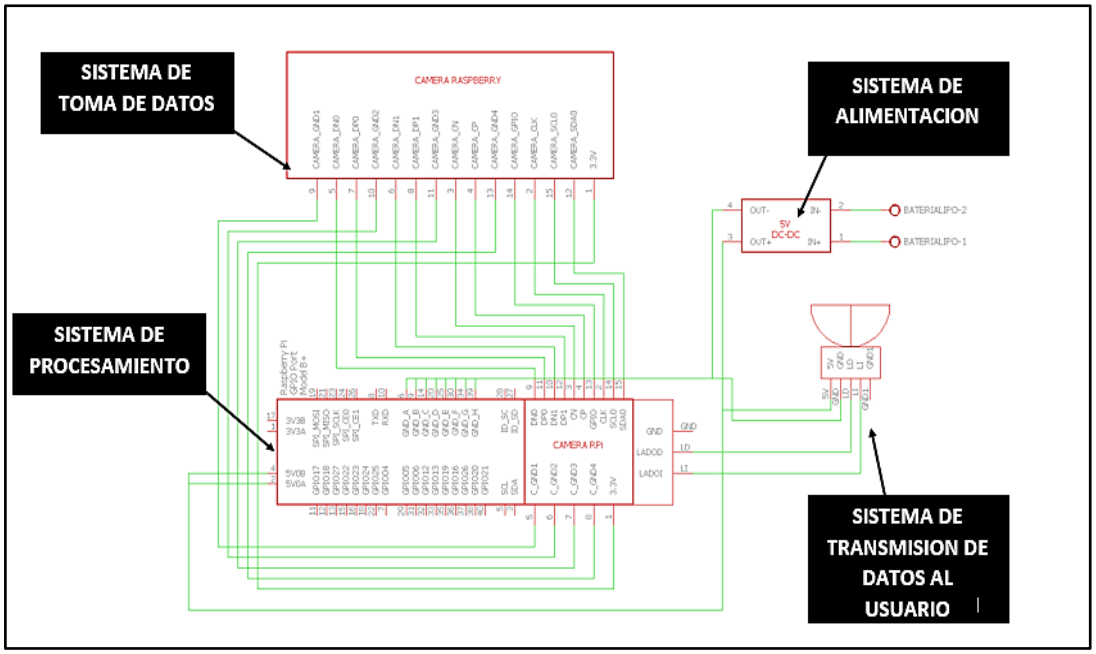


Figura 12-2: Esquema de conexión electrónico

Realizado por: Guillen M., Bonilla T., 2021.

El diseño electrónico se divide en 4 etapas o sistemas descritos en los siguientes apartados.

2.5.2.1 Sistema de alimentación

Sistema encargado de suministrar energía para el funcionamiento adecuado del dispositivo este compuesto por la batería Lipo y circuito STEPDOWN el cual sirve para regular al voltaje de trabajo de los elementos de 5 voltios la ventaja de usar este dispositivo es la gran eficiencia y regulación que posee una eficiencia mayor del 95% garantizando el buen trabajo del sistema.

2.5.2.2 Sistema de transmisión de datos al usuario

Este sistema está conformado básicamente de los parlantes los cuales sirven para enviar la información captada por el algoritmo al usuario final, la alimentación de los parlantes es a 5v.

2.5.2.3 Sistema de procesamiento

Está conformado por la raspberry siendo la parte principal del prototipo ya que realiza la parte de procesamiento de las imágenes y toma de decisiones.

2.5.2.4 Sistema de toma de datos

Es el encargado de tomar las imágenes para su procesamiento por ende está conformado de la cámara No IR. Las conexiones físicas del prototipo no poseen una placa PCB en si ya que casi todos los componentes físicos se conectar por medio de bornera o terminales ya establecidos por poner un ejemplo la cámara la cual tiene un puerto especial conectado mediante un conector y

Flex. A continuación, se detalla las conexiones de los componentes en especial de la raspberry en la tabla 6-2.

Tabla 6-2: Conexión de terminales de la tarjeta Raspberry con los demás elementos

#Ítem	Nombre del terminal	Nombre del terminal del elemento conectado	Descripción
1	DN0	DN0 Cámara	Todos estos pines se conectar por medio de un puerto especial exclusivo de la raspberry mediante un Flex de 15 vías
2	DP0	DP0 Cámara	
3	DN1	DN1 Cámara	
4	DP1	DP1 Cámara	
5	CN	CN Cámara	
6	CP	CP Cámara	
7	GPIO	GPIO Cámara	
8	GPIO	GPIO Cámara	
9	CLK	CLK Cámara	
10	SCL0	SCL Cámara	
11	SDA0	SDA Cámara	
12	C_GND1	GND1 Cámara	
13	C_GND2	GND2 Cámara	
14	C_GND3	GND3 Cámara	
15	C_GND4	GND4 Cámara	
16	Speaker Left	LI	De igual manera esta conexión se realiza mediante el puerto de audio de la raspberry
17	Speaker Right	LD	
18	Speaker GND	GND	
19	5V0A	Out + Step Down	Se conecta a la salida de voltaje del regulador
20	5V0B	Out + Step Down	

Realizado por: Guillen M., Bonilla T., 2021.

2.6 Requerimientos para el diseño del algoritmo

El prototipo debe cumplir los siguientes requerimientos:

- Procesar imágenes en tiempo real.
- Funciones para pre procesar y procesar las imágenes adquiridas.
- Algoritmos y métodos de comparación de imágenes para emitir un resultado.
- Métodos de indexación de imágenes que sirvan como base de datos para comparar imágenes.

2.6.1 Instalación de programas en la Raspberry Pi 4

El primer paso que se realizó fue obtener una tarjeta de memoria Micro SD con capacidad de 350gigabytes clase 10 de la marca SanDisk, eso es lo recomendado por Raspberry con una capacidad de almacenamiento de entre 16 a 32 gigabytes con una velocidad de escritura de 10 MB/s. esta memoria albergara el sistema operativo Raspbian Stretch with Desktop.

2.6.1.1 Sistema operativo Raspbian Stretch

Raspbian es una distribución del sistema operativo GNU/Linux y por lo tanto libre basado en Debian Strech (Debian es una comunidad conformada por desarrolladores y usuarios) para la

placa computadora Raspberry Pi en sus diferentes versiones hasta la fecha presente (ECURED, 2020).

2.6.1.2 Características del sistema operativo Raspbian Stretch

- **Coste:** Debían es un sistema operativo (S.O.) de libre distribución (es decir sin coste alguno).
- **Multiusuario:** permite a varios usuarios acceder al mismo tiempo a través de terminales, y distribuye los recursos disponibles entre todos.
- **Multiplataforma:** Es decir que puede correr en la mayoría de las plataformas del mercado (procesadores de la gama Intel y AMD, Motorola, Sun, Sparc, etc.).
- **Kernel:** Los sistemas Debian actualmente usan el núcleo de Linux.
- **Memoria:** La memoria se gestiona como un recurso unificado para los programas de usuario y para el caché de disco, de tal forma que toda la memoria libre puede ser usada para caché y ésta puede a su vez ser reducida cuando se ejecuten grandes programas.
- **Licencia:** Debian nace como una apuesta por separar en sus versiones el software libre del software no libre, para esto debe respetar 4 libertades: 1. libertad para usarlo.2. libertad para modificarlo.3. libertad para copiarlo.4. libertad para distribuir las modificaciones.
- **Estabilidad:** Como Debian es una distribución que ha probado su estabilidad y utilidad, muchos desarrolladores la han tomado para crear otras nuevas como: Knoppix, Ubuntu, Sidux, etc.
- **Seguridad:** Los problemas de seguridad se solucionan rápidamente con parches de seguridad que se actualizan en internet.

Posteriormente se procedió a descargar el sistema operativo Raspbian de la página oficial, www.raspberrypi.org luego de obtener el sistema operativo, se procedió a formatearlo la memoria bajo FAT-32 que es el formato para manejo de la Raspberry Pi 4. En la figura 13-2 se puede apreciar la imagen del sistema operativo seleccionado a descargar.

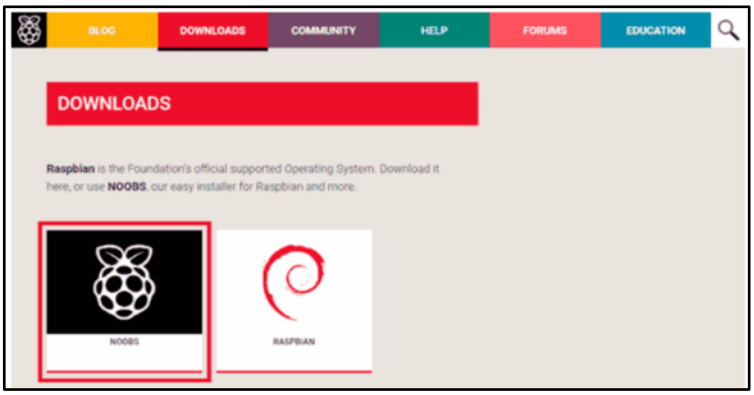


Figura 13-2: Sistema operativo Raspbian a descargar

Fuente: (RASPBERRY PI, 2016c)

2.6.1.3 Instalación del sistema operativo en la Raspberry Pi 4

Luego de la descarga del sistema operativo Raspbian, se procedió a guardar la descarga en la tarjeta de memoria Micro SD, mediante el programa Win32 Disk Imager, el cual permitió seleccionar el archivo de imagen y la ubicación del dispositivo en donde se desea guardar la descarga correspondiente, la cual fue descargada en la tarjeta SD. En la Figura 14-2 se muestra la pantalla principal del programa Win32 Disk Imager.

Figura 14-2: Programa Win 32 Disk Imager

Realizado por: Guillen M., Bonilla T., 2021.

Con esto la imagen almacenada en la tarjeta Micro SD se conectó a la Raspberry junto con todos los periféricos de entrada y salida (teclado, pantalla, mouse y cable ethernet), para el manejo de la placa. Al energizar la Raspberry automáticamente arrancó el sistema operativo y mostró la pantalla de inicio del escritorio, como en la Figura 15-2.

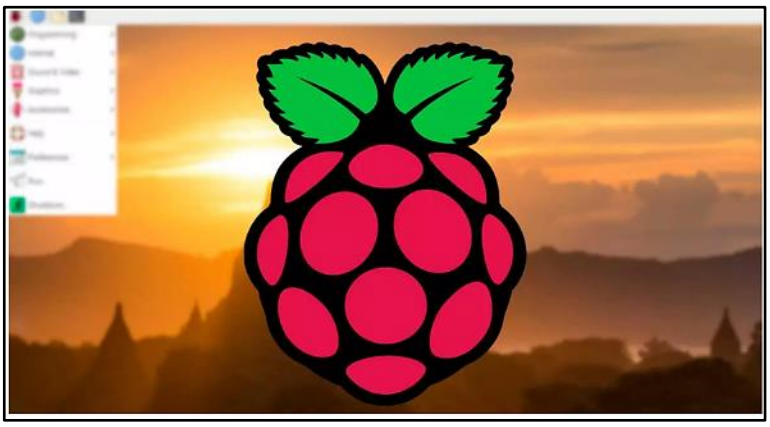


Figura 15-2: Pantalla de escritorio de la Raspberry Pi 4

Realizado por: Guillen M., Bonilla T., 2021.

2.6.1.4 Descarga e instalación de Python 3.0 y sus librerías

Después de que el sistema operativo funcionara correctamente, se pasó a la instalación de las librerías usadas en este prototipo, que permitieron realizar tareas de visión artificial por computador. De acuerdo con la guía de instalación de OpenCV por parte de pyimagesearch, que se la puede encontrar en la siguiente dirección web, https://www.pyimagesearch.com/2017/09/04 /raspbian-stretch-install-opencv-3-python-on-your-raspberry-pi/ explica paso a paso la

instalación y además se pueden encontrar otras guías, para la instalación de otras librerías complementarias.

2.6.1.5 Librerías por utilizarse en la programación del prototipo

Una vez realizado todo el proceso anterior procedemos con la instalación de las librerías y complementos, con la cual se van a ir desarrollando la programación para el funcionamiento de nuestro prototipo.

Las librerías implícitas en este programa están entrelazadas más con el procesamiento de imágenes, así como el manejo de matrices, la primera y la principal es OpenCV con ella se realizó la programación en su mayoría, ya que este abre un conjunto de funciones y brinda una gran facilidad al momento de programar. Una de las principales tareas de esta función es la utilización de la segmentación mediante el uso de la detección de contornos y la conversión a distintos formatos de color, como se puede apreciar los dos iconos de la Figura 16-2

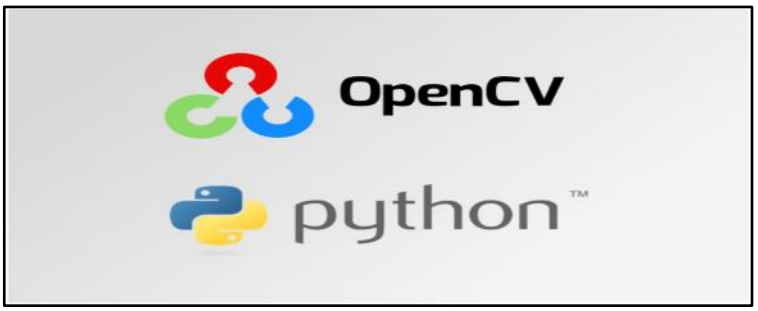


Figura 16-2: OpenCV y Python

Realizado por: Guillen M., Bonilla T., 2021.

La segunda herramienta es NumPy, esta librería abre un conjunto de funciones que permiten crear insertar, eliminar y seleccionar elementos en una matriz, así como también hacer análisis estadísticos básicos como la media la mediana y la varianza, también la conversión de un vector a matriz y viceversa. El logo de esta librería se muestra en la figura 17-2.

Figura 17-2: Librería NumPy

Realizado por: Guillen M., Bonilla T., 2021.

La tercera librería es gTTs es un complemento de OpenCV que usa la API de Google (Google Text-to-Speech) para convertir el texto a audio, esta se muestra en la figura 18-2.

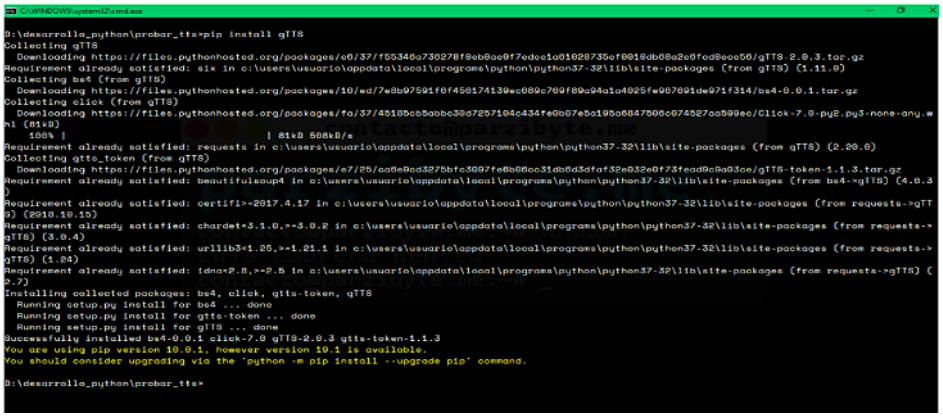


Figura 18-2: Librería gTTs

Realizado por: Guillen M., Bonilla T., 2021.

La cuarta librería mostrada en la figura 19-2 es matplotlib es una biblioteca para la generación de gráficos a partir de datos contenidos en listas o arrays en el lenguaje de programación Python y su extensión matemática NumPy.

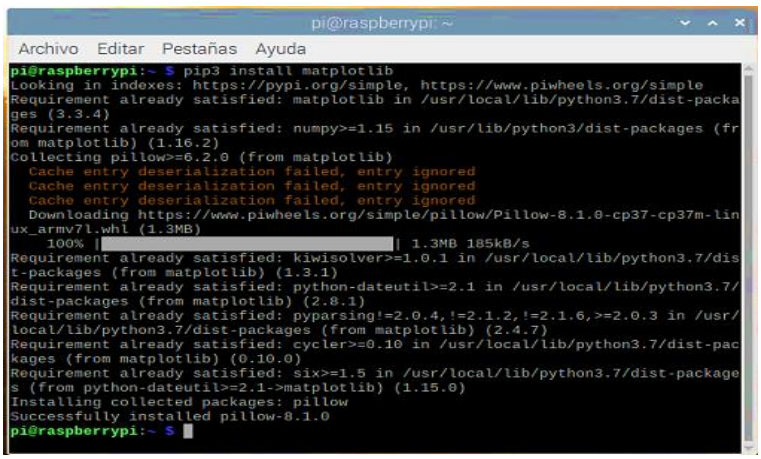


Figura 19-2: Librería matplotlib

Realizado por: Guillen M., Bonilla T., 2021.

La quinta librería mostrada en la figura 20-2 es PiCamera esta opción nos permite visualizar en tiempo real la imagen de la cámara, ademas de poder ajustar los parámetros como brillo y ver los resultados inmediatamente en la vista previa. Crearemos un ejemplo que nos permita ver la vista previa.

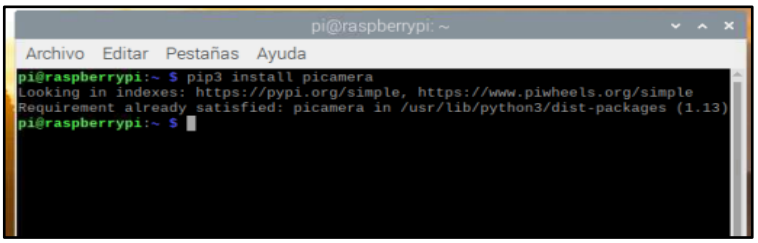


Figura 20-2: PiCamera

Realizado por: Guillen M., Bonilla T., 2021.

2.6.2 Entrenamiento del algoritmo HAAR Cascade

La realización de este se muestra en el siguiente aparatado.

2.6.2.1 Algoritmo de entrenamiento

Para el entrenamiento del algoritmo Cascade se utiliza la guide Cascade Trainer GUI. La pantalla principal del entrenador se indica en la figura 21-2.

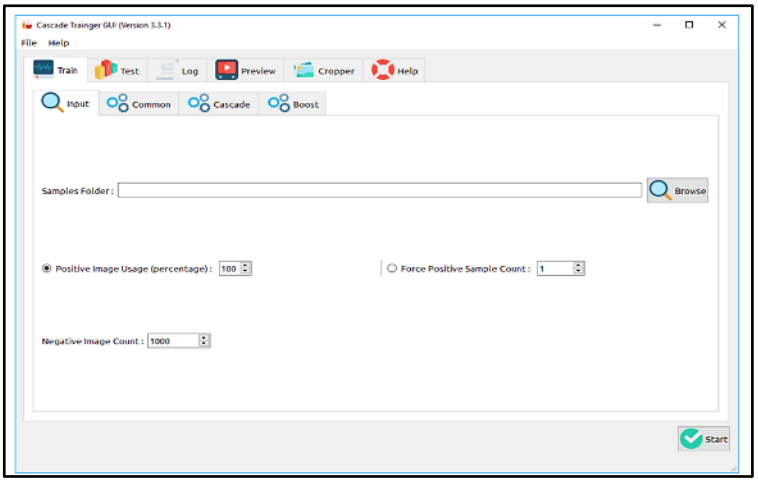


Figura 21-2: Pantalla principal del Cascade Trainer GUI

Realizado por: Guillen M., Bonilla T., 2021.

El entrenador facilita y reduce el número de pasos comparado con otras formas de entrenamiento, para esto se cumplen los siguientes pasos:

- **Primer paso:** Recopilación de información (imágenes o videos), en este caso las imágenes a ser reconocidas pueden ser en forma de video o fotografía para mayor facilidad en la selección de imágenes se prefiere tomar videos de los objetos como se muestra en la figura 22-2.

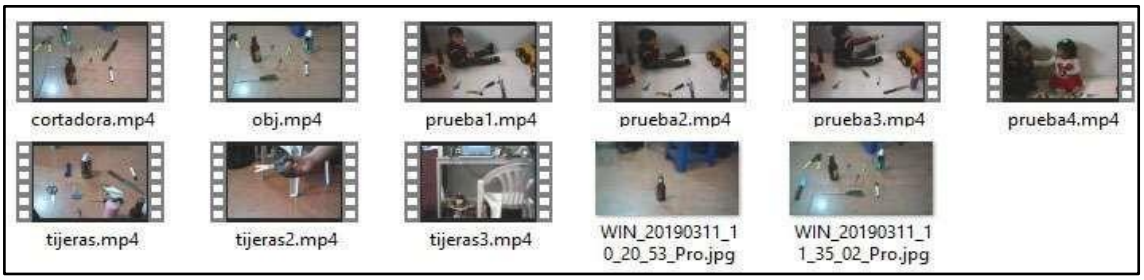


Figura 22-2: Recopilación de imágenes y videos

Realizado por: Guillen M., Bonilla T., 2021.

- **Segundo paso:** Se procede a seleccionar los objetos positivos de las imágenes o videos que se copilaron anteriormente, la aplicación Cascade Trainer GUI además de poseer el entrenador también tiene una aplicación que sirve para seleccionar las imágenes positivas u

objetos de los videos o imágenes grabadas. Dicha aplicación se puede usar para imágenes positivas o negativas como se observa en la figura 23-2. Como se muestra en la figura es necesario seleccionar la imagen positiva o negativa la cual se va a guardar en una carpeta ya antes seleccionada.

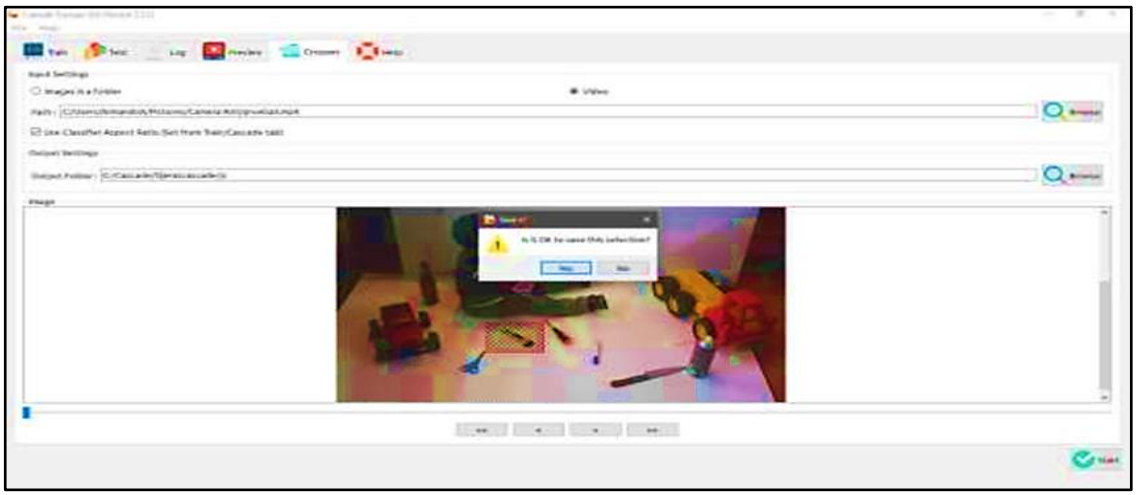


Figura 23-2: Selección de imágenes positivas

Realizado por: Guillen M., Bonilla T., 2021.

- **Tercer paso:** Después de seleccionar o guardar las imágenes positivas o negativas en las carpetas correspondientes como se muestra en la figura 24-2 se procede aumentar el número de imágenes negativas desde el HAAR Cascade Master desde la base de datos que ya tiene la aplicación por defecto.

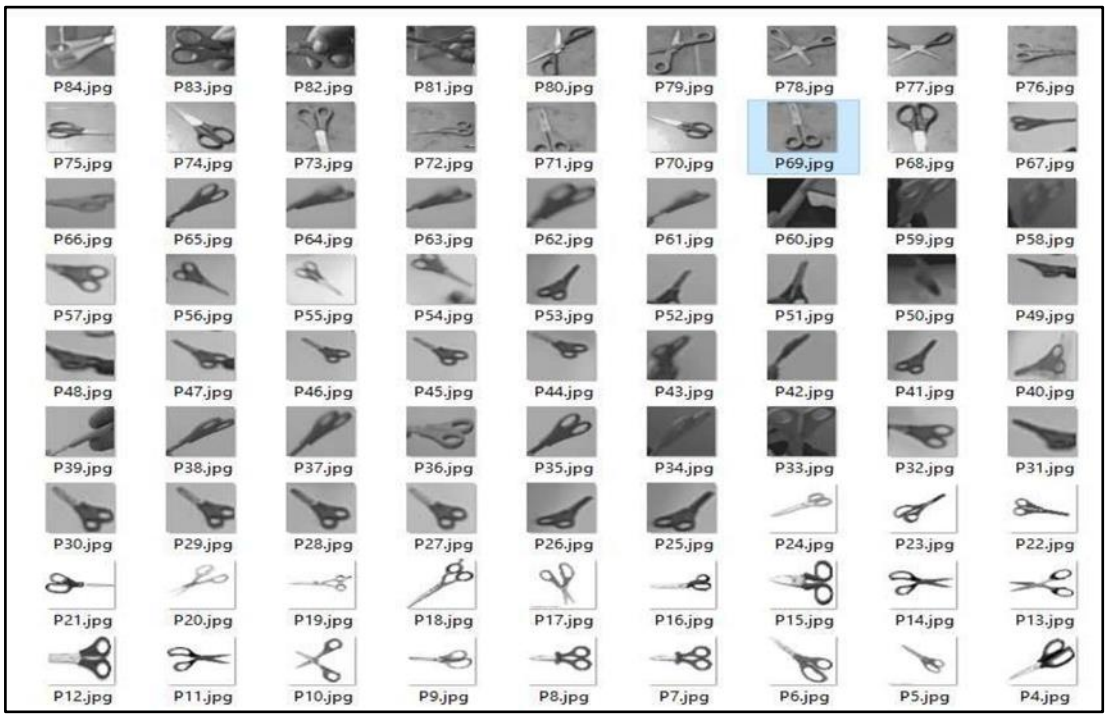


Figura 24-2: Imágenes positivas guardadas

Realizado por: Guillen M., Bonilla T., 2021.

- **Cuarto paso:** Se configura el entrenador, por ejemplo, el número de stages, la memoria RAM del entrenador y el número de imágenes positivas y negativas, así como otros parámetros más importantes para el entrenamiento. Esta configuración se muestra en la figura 25-2. Después de configurar se procede a la inicialización, el tiempo del entrenamiento depende del dispositivo que se está utilizando para este caso se empleó una portátil HP Envy 17" con procesador Core i7 y 16 Gb de memoria RAM.

Figura 25-2: Configuración del entrenador

Realizado por: Guillen M., Bonilla T., 2021.

- **Quinto paso:** Final del entrenamiento se visualiza en la figura 26-2, después de un determinado tiempo de entrenamiento dependiendo del número de imágenes que en este caso fue de 350 imágenes positivas y 1700 imágenes negativas aproximadamente tuvo un tiempo de entrenamiento de 5.40 horas al finalizar el Cascade se obtienen Stages que se han probado en el entrenamiento y un archivo XML que se utilizará en la programación. La creación ese archivo se visualiza en la imagen 27-2.

Figura 26-2: Final del entrenamiento

Realizado por: Guillen M., Bonilla T., 2021.

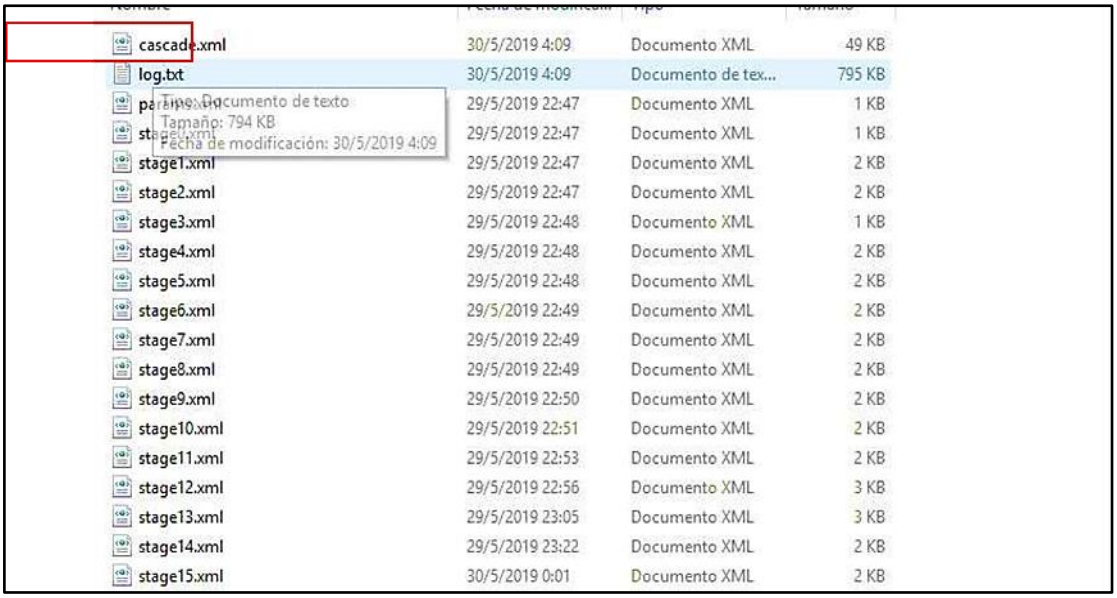


Figura 27-2: Archivo XML generado al final

Realizado por: Guillen M., Bonilla T., 2021.

2.7 Comprobación del entrenamiento

Para comprobar si el Cascade tuvo éxito en el entrenamiento en la guide Cascade Trainer GUI tiene su propio comprobador de Cascade el cual permite subir videos o imágenes configurando el tamaño de las muestras y los vecinos más cercanos. La salida es un video o conjunto de imágenes. Esto se muestra en la figura 28-2.

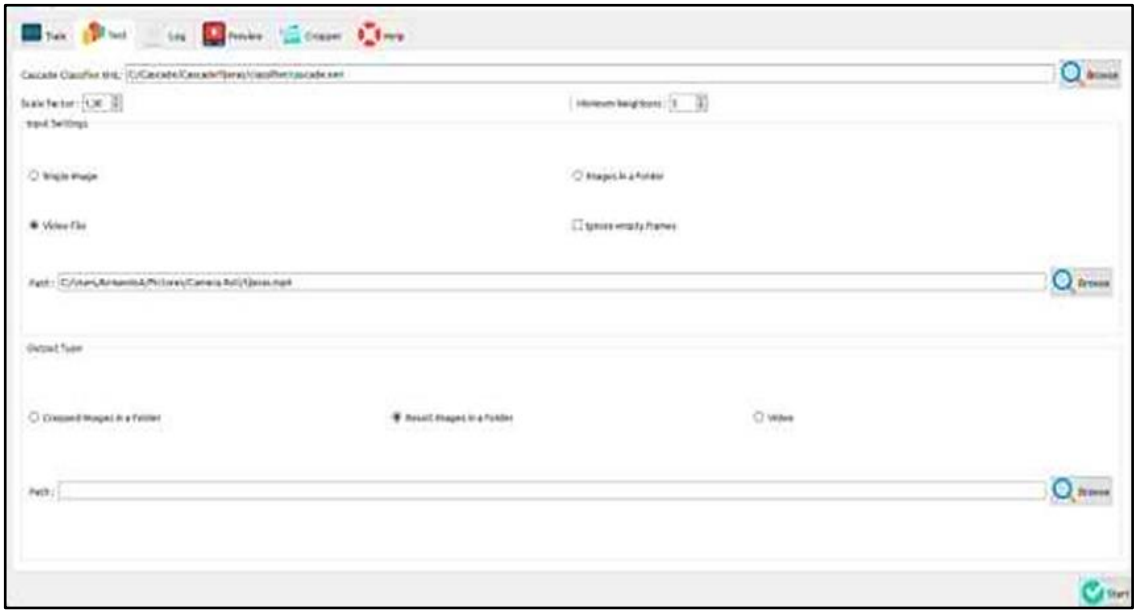


Figura 28-2: Comprobación del entrenamiento

Realizado por: Guillen M., Bonilla T., 2021.

2.8 Programación en Python

En las siguientes secciones se da a conocer las etapas consideradas para la programación del prototipo en Python.

2.8.1 Algoritmo de la programación en Python

Compuesta por el diagrama de flujo en el que se indica el orden de las instrucciones a ejecutarse, esto se visualiza en la figura 29-2.

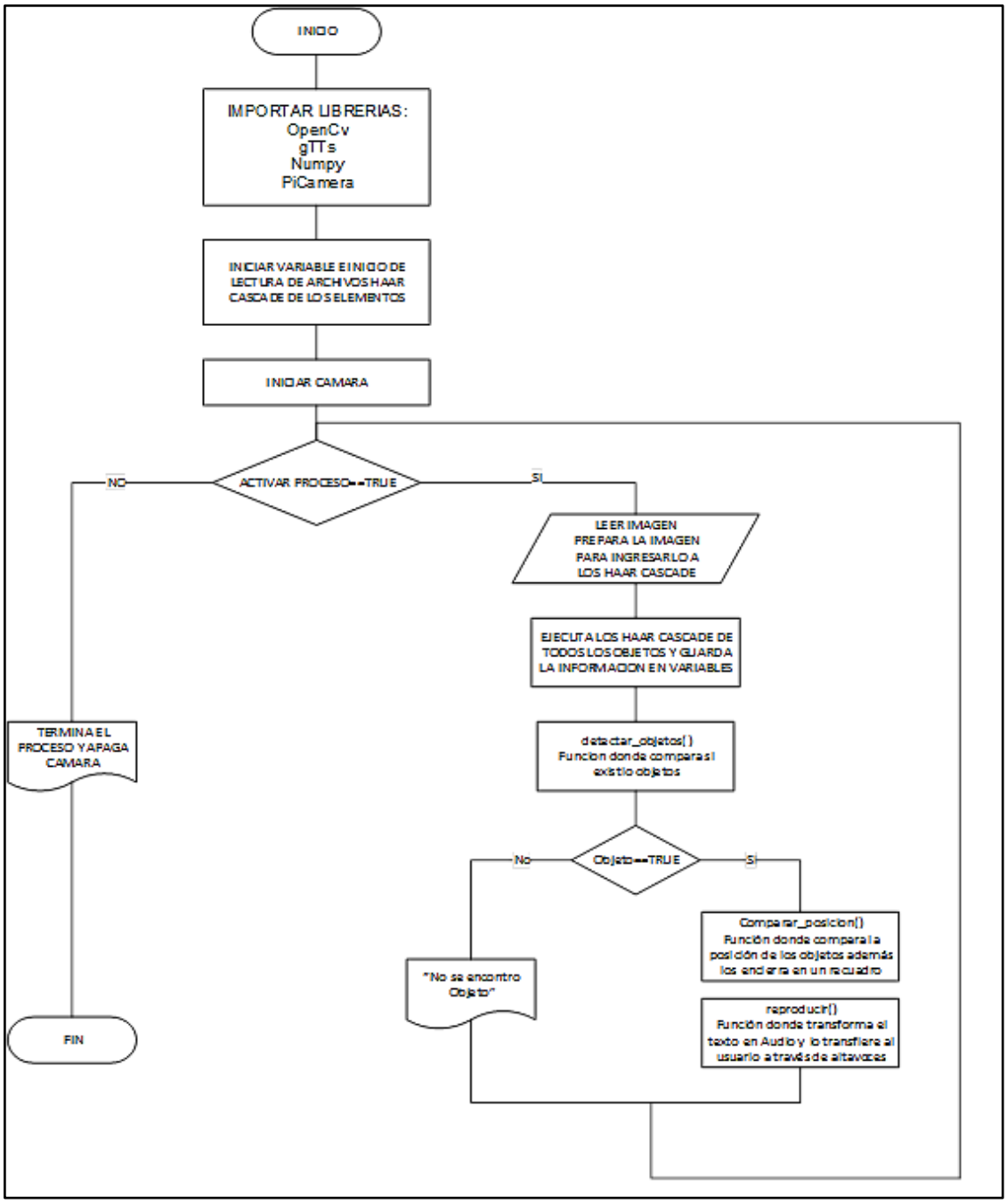


Figura 29-2: Algoritmo de la programación en Python

Realizado por: Guillen M., Bonilla T., 2021.

Como primera parte tenemos el inicio de librerías de Python como son la librería OpenCv, NumPy, PiCamera como las librerías principales además el inicio de variables y la lectura de los archivos XML creados en la etapa de entrenamiento figura 30-2.

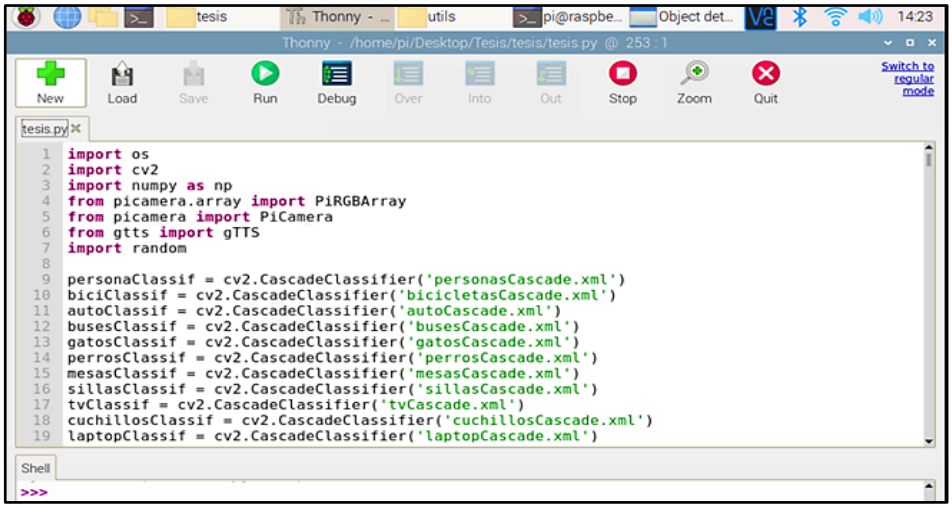


Figura 30-2: Programación Python - Librerías

Realizado por: Guillen M., Bonilla T., 2021.

A continuación, iniciamos la cámara y además dividimos en sectores la pantalla para poder saber dónde está ubicado los distintos objetos para ello se divide en 6 partes de posicionamiento como se muestra en la figura 31-2.

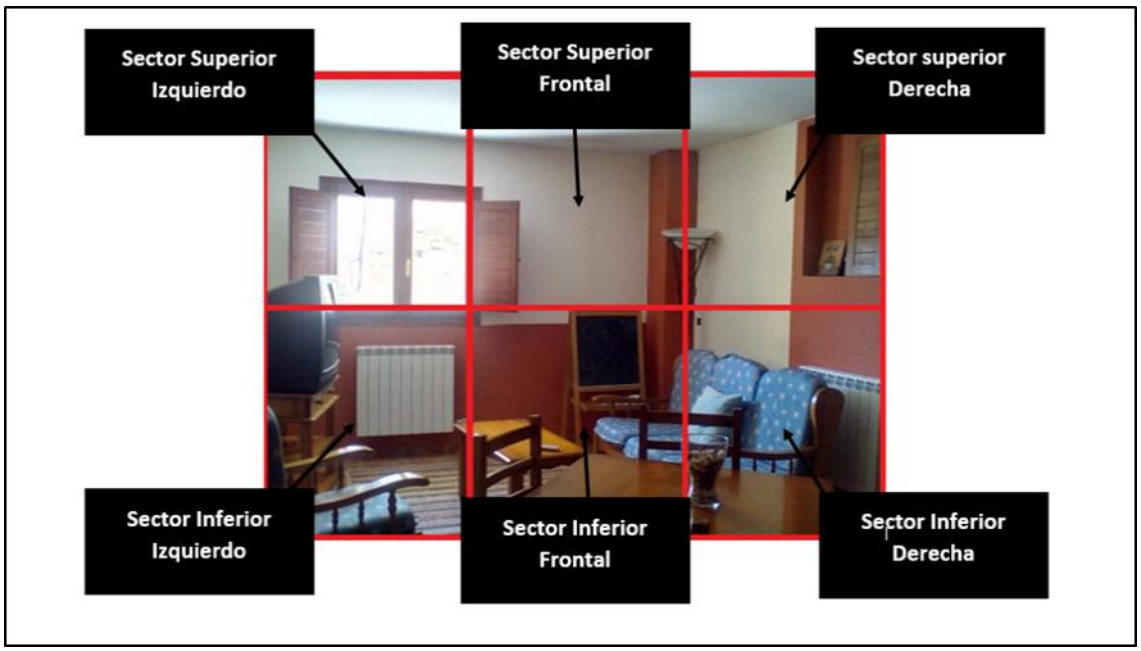


Figura 31-2: Cámara y división de sectores

Realizado por: Guillen M., Bonilla T., 2021.

Después de estas divisiones se configura a la imagen y para esto se usó el código mostrado en la figura 32-2.

```
frame=camara.capture_continuous(imagen, format="bgr")
ancho = int(frame.get(cv2.CAP_PROP_FRAME_WIDTH))
alto = int(frame.get(cv2.CAP_PROP_FRAME_HEIGHT))
a = math.ceil(ancho * (0))
b = math.ceil(ancho * (0.33))
c = math.ceil(ancho * (0.66))
d = math.ceil(ancho * (1))
A = math.ceil(alto * (0))
A = math.ceil(alto * (0.5))
A = math.ceil(alto * (1))
```

Figura 32-2: Configuración de la imagen

Realizado por: Guillen M., Bonilla T., 2021.

Después de esta primera etapa compara si el activador del proceso esta verdadero o no, si es verdadero empieza a realizar el proceso si es falso sale del sistema. El proceso comienza cuando ejecuta todos los HAAR Cascade de todos los objetos uno a la vez como se muestra en la figura 33-2, el código para ejecutar el HAAR Cascade y los parámetros que debemos configurar.

```
personas = personasClassif.detectMultiScale(gray,scaleFactor = 5
                                            ,minNeighbors = 91,minSize=(70,78))
```

Figura 33-2: Código HAAR Cascade

Realizado por: Guillen M., Bonilla T., 2021.

Donde:

- **Gray:** Es la imagen en donde va a actuar la detección.
- **ScaleFactor:** Este parámetro especifica que tanto se reducirá la imagen. Hay que tener en cuenta si damos un número muy alto, se pierden algunas detecciones. Mientras que para valores muy pequeños llevará mayor tiempo de procesamiento.
- **MinNeighbors:** Este parámetro especifica cuántos vecinos debe tener cada rectángulo candidato para retenerlo.
- **MinSize:** Este parámetro indica el tamaño mínimo posible del objeto.
- **MaxSize:** Este parámetro indica el tamaño máximo posible del objeto.

```
frame=camara.capture_continuous(imagen, format="bgr")
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
if(valorI==1):
    personas = personasClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,mi
if(valorI==2):
    bici = biciClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(7
if(valorI==3):
    auto = autoClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(7
if(valorI==4):
    buses = busesClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=
if(valorI==5):
    gatos = gatosClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=
if(valorI==6):
    perros = perrosClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSiz
if(valorI==7):
    mesas = mesasClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=
if(valorI==8):
    sillas = sillasClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSiz
```

Figura 34-2: Extracción de objetos

Realizado por: Guillen M., Bonilla T., 2021.

Como resultado obtenemos una matriz que proporción la posición del objeto dado en pixeles, los datos para ser específicos son la posición X, Y del objeto el punto de origen además del ancho y alto en pixeles del objeto. Esto se visualiza en la figura 35-2. Una vez terminado de ejecutar todos los HAAR Cascade entra en sistema o algoritmo compara si existió o no un objeto en este caso la función en la programación que realiza este proceso se llama detect_obj () más adelante se realizara el análisis de cada función.

Figura 35-2: Comparación de funciones

Realizado por: Guillen M., Bonilla T., 2021.

Después de la función de detección el algoritmo pregunta si detecto algún objeto, si es verdadero corre la función compara posición de los objetos con los 6 sectores antes mencionados además de guardar la posición en texto de los objetos. Para finalizar la librería gTTs efectúa el algoritmo OCR para transformar el texto almacenado en audio el cual será transmitido al usuario final mediante parlantes.

2.8.2 Funciones del algoritmo implementado

En el siguiente apartado se indica el diagrama de flujo de las funciones empleadas en la implementación del algoritmo del prototipo. Compuesto por varios procesos de lectura, ejecución, de decisión y bucles repetitivos.

2.8.2.1 Función detect_obj ( )

A continuación, se visualiza cada una de las acciones en base al uso de esta función. Esto se puede observar en la figura 36-2.

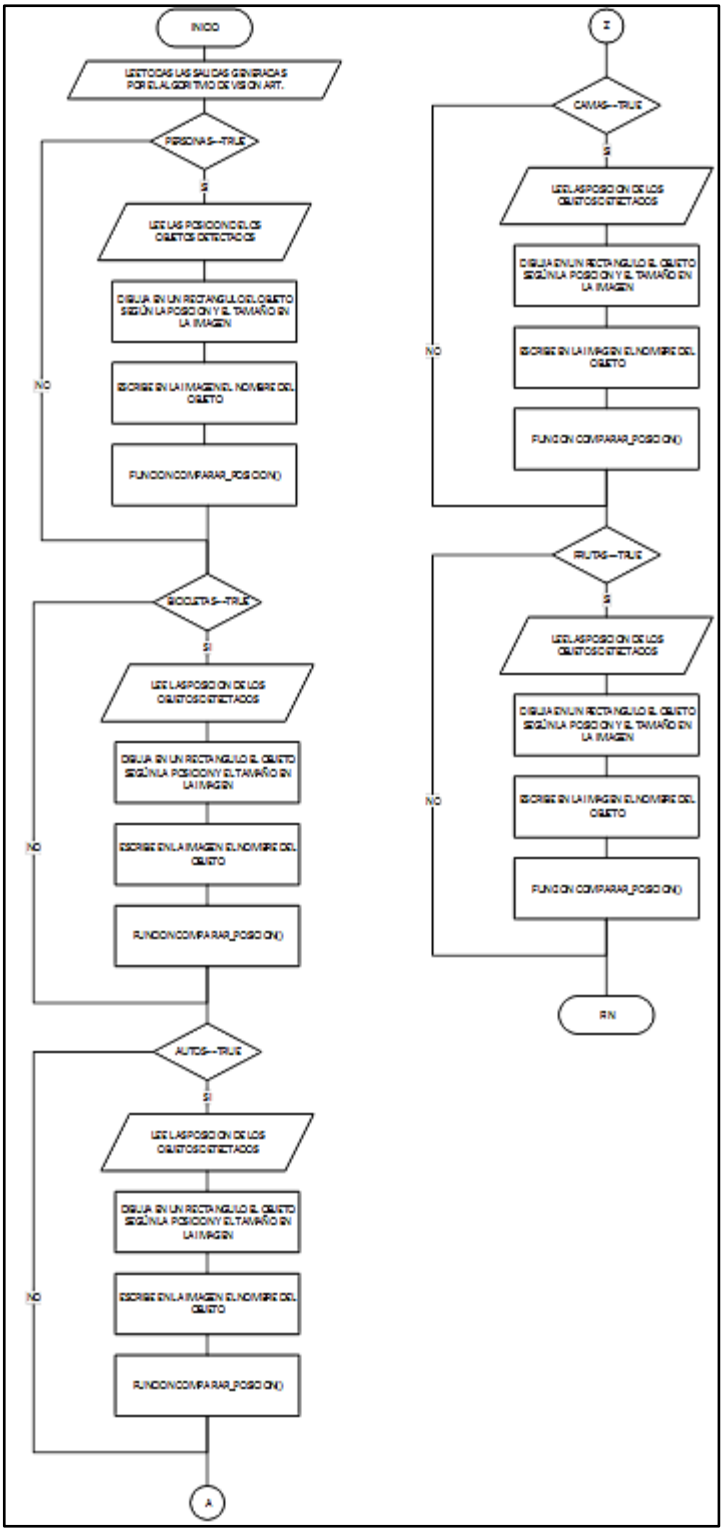


Figura 36-2: Función detect ( )

Realizado por: Guillen M., Bonilla T., 2021.

Dentro de esta función se compara los resultados del HAAR Cascade por ejemplo el objeto personas si fue detectado el algoritmo dibuja un rectángulo alrededor del objeto según la posición y tamaño obtenidos del algoritmo de detección leídos en las variables x, y, w, h además escribe su nombre en la parte superior. Esto se muestra en la figura 37-2.

Además, se tiene una lista de colores predefinidos los cuales se escogen en forma aleatorio para cada objeto después de señalar los objetos se calcula el punto medio para enviar esta información a la función encargada de detectar en el sector que se encuentra el objeto.

Cabe recalcar que este proceso se repite el número de objetos entrenados en este caso 20 objetos, pero por cuestión de espacio la imagen de la función anterior se redujo para poder explicar su funcionamiento.

```
for (x,y,w,h) in personas:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,objetos[1],(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, y-10), (x + 20, y + 10), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,1)
```

Figura 37-2: Resultados de HAAR Cascade

Realizado por: Guillen M., Bonilla T., 2021.

2.8.2.2 Función comparar_posición ( )

A continuación, se visualiza cada una de las acciones en base al uso de esta función. Esto se puede observar en la figura 38-2.

```
def comparar_posicion(xcentro, ycentro,numero):
    activador=True
    if(xcentro>a and xcentro<=b):
        if(ycentro>A and ycentro<B):
            texto=texto+objeto[numero]+pos1
        else:
            texto=texto+objeto[numero]+pos2

     if(xcentro>b and xcentro<=c):
        if(ycentro>A and ycentro<B):
            texto=texto+objeto[numero]+pos3
        else:
            texto=texto+objeto[numero]+pos4

     if(xcentro>c and xcentro<=d):
        if(ycentro>A and ycentro<B):
            texto=texto+objeto[numero]+pos5
        else:
            texto=texto+objeto[numero]+pos6
```

Figura 38-2: Función comparar ( )

Realizado por: Guillen M., Bonilla T., 2021.

En esta función primero lee la posición del objeto y el nombre después entra en una etapa de comparación anidada en la cual pregunta en que sector se encuentra el objeto cuando se detecta el sector en una variable tipo texto se almacena el nombre y el nombre del sector cabe recalcar que se tiene 6 sectores.

- Sector Superior Derecha
- Sector Superior Frontal

- Sector Superior Izquierda
- Sector Inferior Derecha
- Sector Inferior Frontal
- Sector Inferior Izquierda

2.8.2.3 Función reproducir ( )

A continuación, se observa cada una de las acciones en base al uso de esta función. Esto se puede visualizar en la figura 39-2.

Figura 39-2: Función reproducir ( )

Realizado por: Guillen M., Bonilla T., 2021.

Esta función se ejecuta al final de la detección y de la localización de los objetos, primero lee el archivo de texto creado anteriormente donde está la información de los objetos y su posición, después el algoritmo OCR transforma el texto en audio esta información se guarda en un archivo .mp3 el cual es reproducido después mediante los parlantes. Esta sección de la programación se muestra en la figura 40-2.

```
def reproducir():
    sonido = gTTS(text lang="es", slow=False)
    sonido.save("objeto.mp3")
    os.system("mpg321 objeto.mp3")
```

Figura 40-2: Código de la función reproducir

Realizado por: Guillen M., Bonilla T., 2021.

CAPÍTULO III

3. VALIDACIÓN DEL PROTOTIPO

En este capítulo se presentaron las pruebas y los resultados del prototipo construido. Donde se evaluó el algoritmo HAAR Cascade para la detección de objetos y su funcionamiento, se realizaron pruebas con imágenes de caracteres de diferentes objetos, donde posteriormente fueron evaluadas con imágenes reales, en la cual el algoritmo debe evaluar si existen dichos objetos para emular al prototipo.

3.1 Tiempo de respuesta del algoritmo

Para determinar el tiempo de respuesta del algoritmo o la ejecución de este, se usó la función time( ) del módulo estándar que lleva el mismo nombre (el módulo time), esta función nos retorna un valor de tipo flotante con el tiempo de ejecución del algoritmo, esta es una función de Python, permite medir el tiempo transcurrido durante la ejecución de un determinado código.

El tiempo de respuesta del algoritmo es de segundos en promedio de 0,61039. Esto se puede visualizar en la figura 1-3.

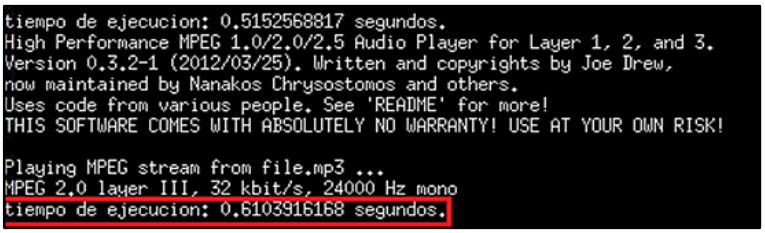


Figura 1-3: Detección de objetos

Realizado por: Guillen M., Bonilla T., 2021.

3.2 Desarrollo de pruebas al prototipo

A continuación, se detallan las pruebas que se realizaron al prototipo para comprobar que los objetivos establecidos sean cumplidos y los resultados sean los esperados. Para ello se evalúan cada una de las señales de manera individual que en total son 8 el número de pruebas escogido es alto para tener resultados más precisos y el porcentaje de validación del prototipo sea alto.

3.2.1 Experimento de validación del prototipo implementado

Para la validación del prototipo se realizaron las siguientes pruebas:

- Objeto persona entre 3 y 2 metros con un rango entre 1500 y 500 lux
- Objeto Cama entre 3 y 2 metros con un rango entre 1500 y 500 lux

- Objeto Perro entre 1 y 2 metros con un rango entre 1500 y 500 lux
- Objeto Cama y silla entre 2 y 3 metros con un rango entre 1500 y 500 lux
- Objeto Silla y Celular entre 2 y 3 metros con un rango entre 1500 y 500 lux
- Objeto silla entre 2 y 3 metros con un rango entre 500 y 80 lux
- Objeto silla entre 1 y 2 metros con un rango entre 500 y 80 lux
- Objeto Televisión entre 1 y 2 metros con un rango entre 500 y 80 lux

Objeto persona entre 2 y 3 metros con un rango entre 1500 y 500 lux, esta detección es visualizada en la figura 2-3.

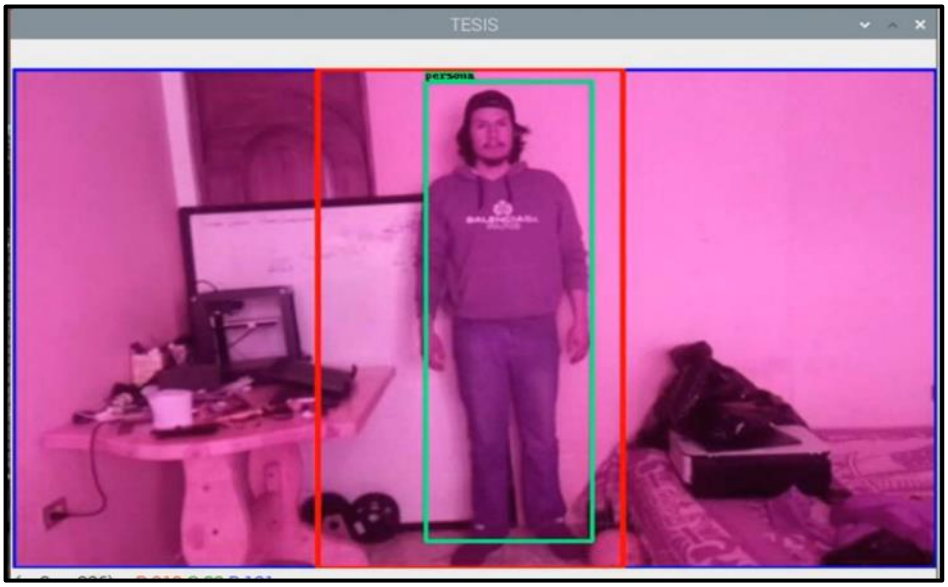


Figura 2-3: Detección del objeto persona

Realizado por: Guillen M., Bonilla T., 2021.

Objeto cama entre 2 y 3 metros con un rango entre 1500 y 500 lux, esta detección es visualizada en la figura 3-3.

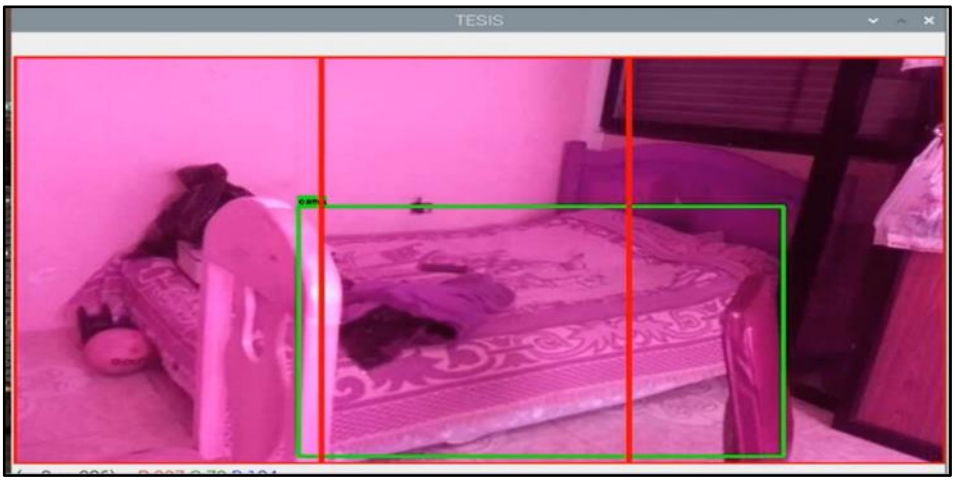


Figura 3-3: Detección del objeto cama

Realizado por: Guillen M., Bonilla T., 2021.

Objeto perro entre 1 y 2 metros con un rango entre 1500 y 500 lux, esta detección es visualizada en la figura 4-3.

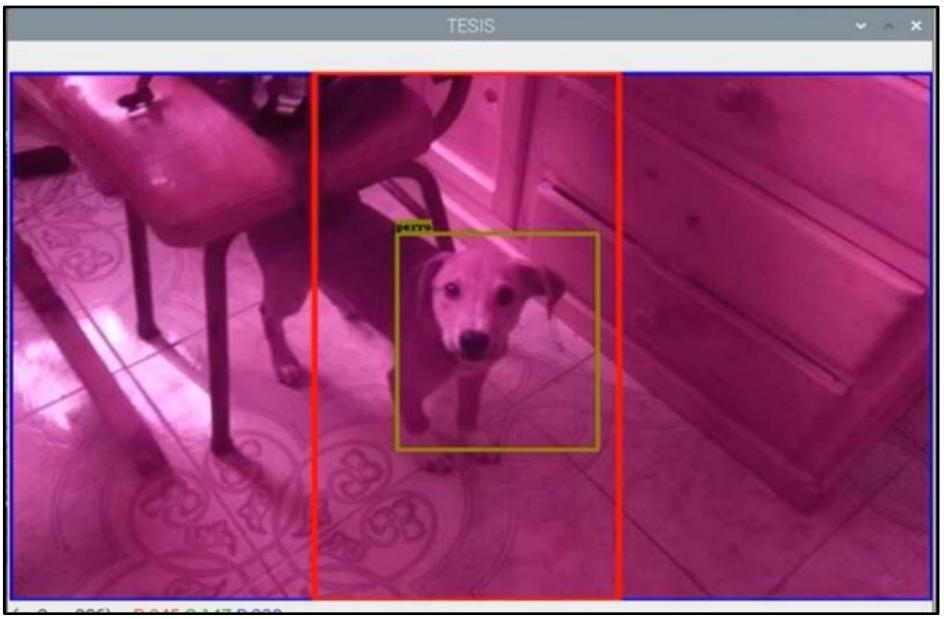


Figura 4-3: Detección del objeto perro

Realizado por: Guillen M., Bonilla T., 2021

Objeto cama y silla entre 2 y 3 metros con un rango entre 1500 y 500 lux, esta detección es visualizada en la figura 5-3.

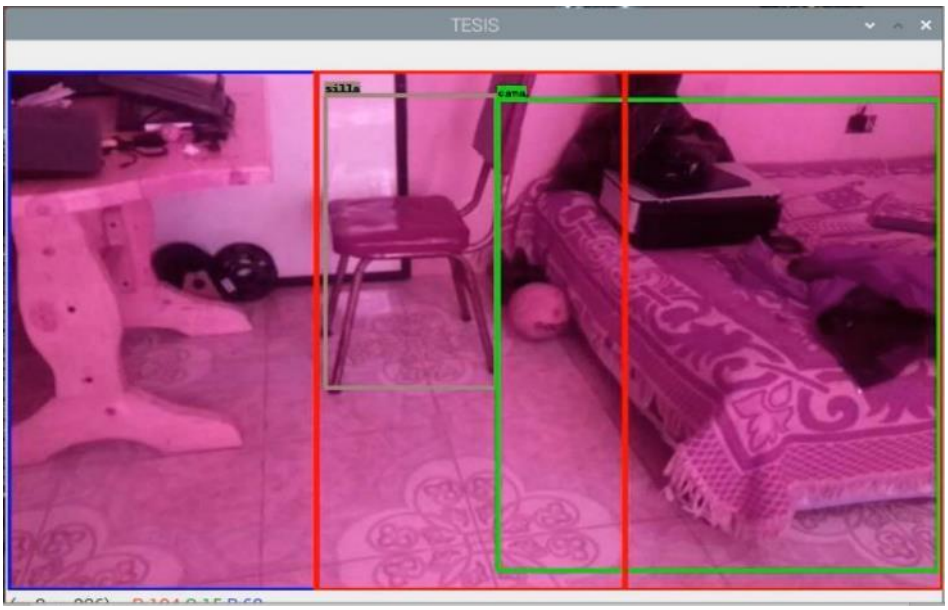


Figura 5-3: Detección del objeto cama y silla

Realizado por: Guillen M., Bonilla T., 2021

Objeto silla y celular entre 3 y 2 metros con un rango entre 1500 y 500 lux, esta detección es visualizada en la figura 6-3.

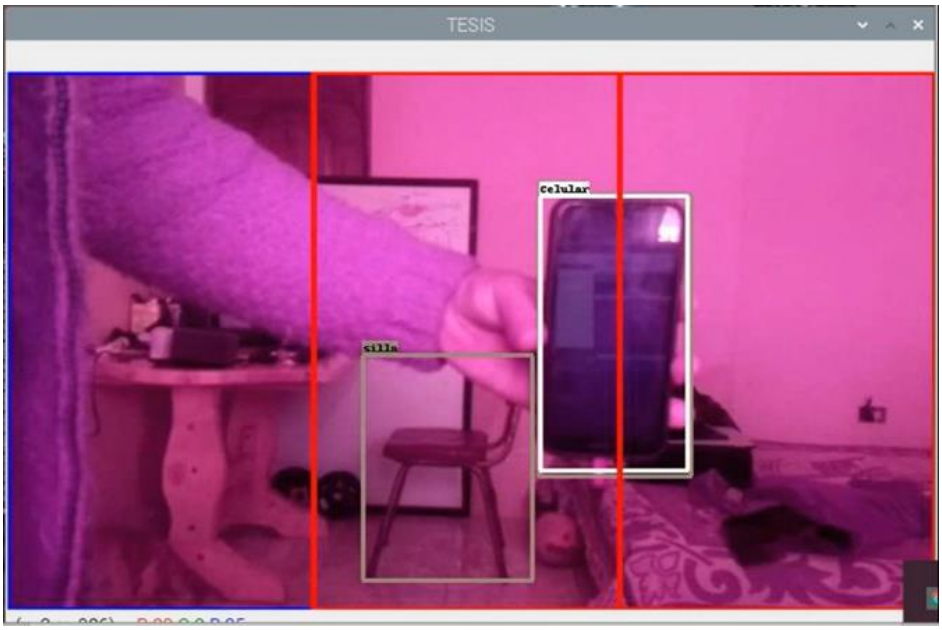


Figura 6-3: Detección del objeto silla y celular

Realizado por: Guillen M., Bonilla T., 2021

Objeto silla entre 2 y 3 metros con un rango entre 500 y 80 lux, esta detección es visualizada en la figura 7-3.

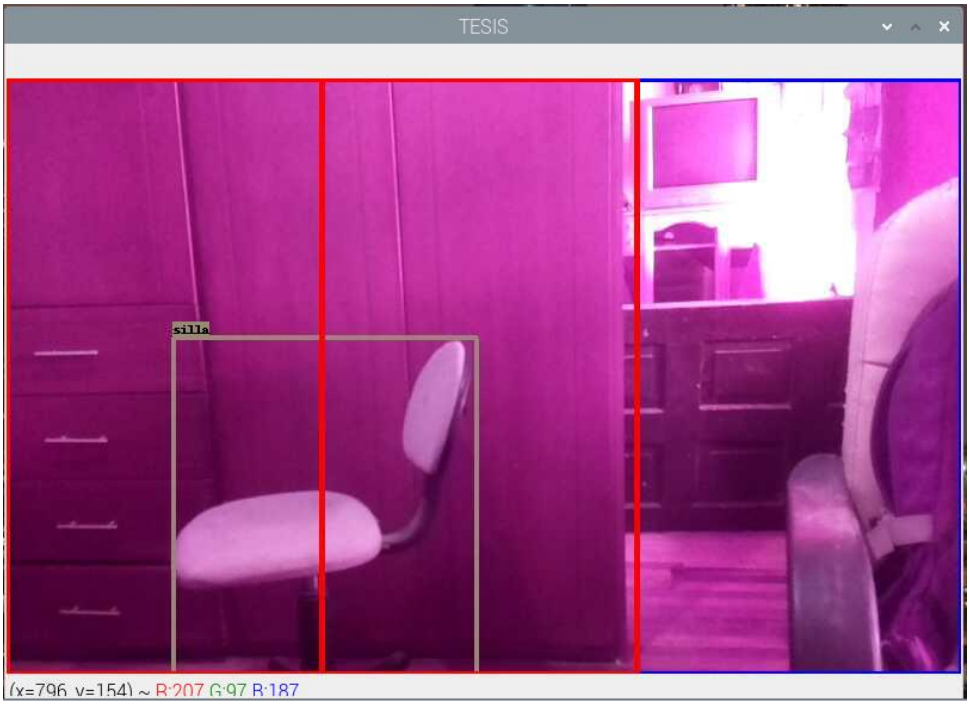


Figura 7-3: Detección del objeto silla

Realizado por: Guillen M., Bonilla T., 2021

Objeto cuchillo entre 1 y 2 metros con un rango entre 500 y 80 lux, esta detección es visualizada en la figura 8-3.

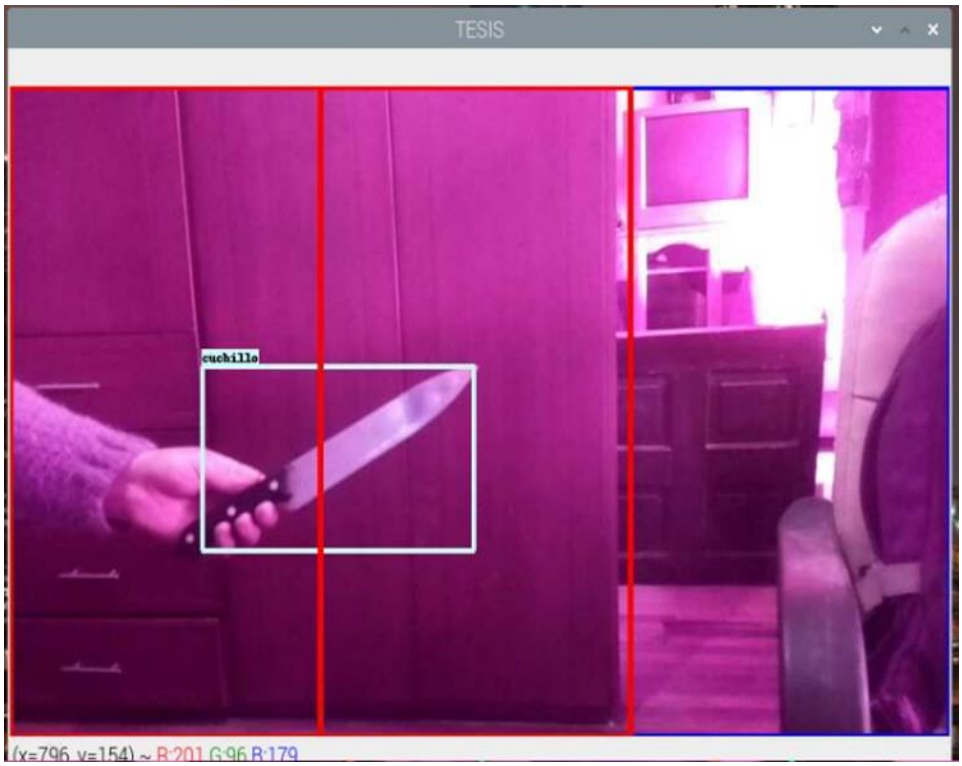


Figura 8-3: Detección del objeto cuchillo

Realizado por: Guillen M., Bonilla T., 2021

Objeto libro entre 1 y 2 metros con un rango entre 500 y 80 lux, esta detección es visualizada en la figura 9-3.

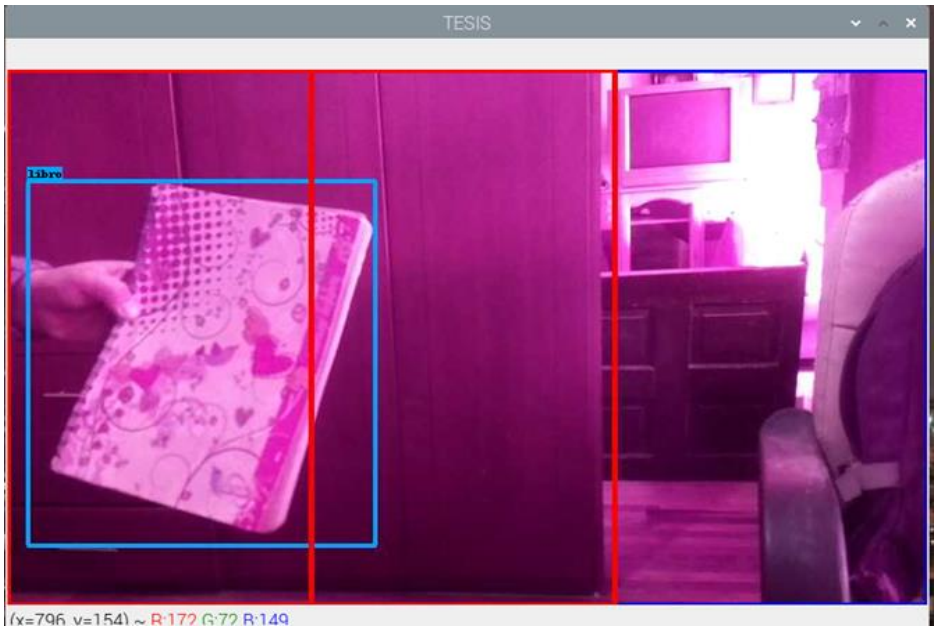


Figura 9-3: Detección del objeto libro

Realizado por: Guillen M., Bonilla T., 2021

Objeto televisión entre 1 y 2 metros con un rango entre 500 y 80 lux , esta detección es visualizada en la figura 10-3.

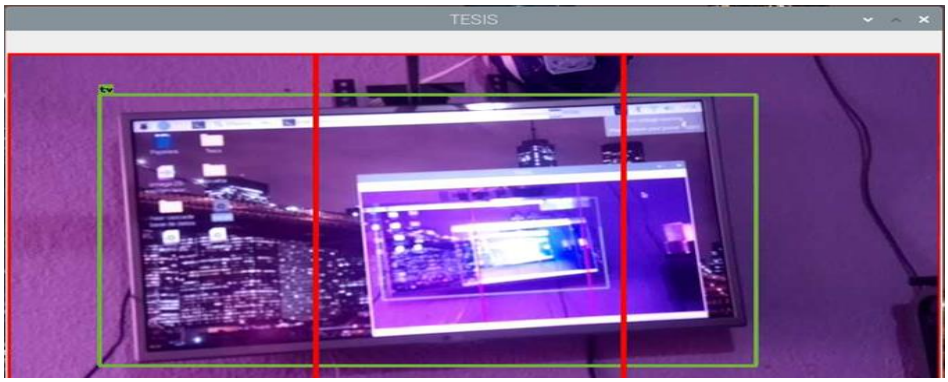


Figura 10-3: Detección del objeto televisión

Realizado por: Guillen M., Bonilla T., 2021

Para las pruebas se realizar con dos criterios especialmente:

- Cantidad de luz (Lux)
- Distancia del objeto a la Cámara

Para las pruebas se tomaron 3 tipos de rango con una variación de luxes considerada:

- Rango 1 entre 5000 y 1500 lux para el cálculo del valor se instaló la aplicación “LUX” la cual nos indica el nivel de lux en tiempo real.
- Rango 2 entre 1500 y 500 lux
- Rango 3 entre 400 y 50 lux

La cantidad de Lux se indica en la figura 11-3.

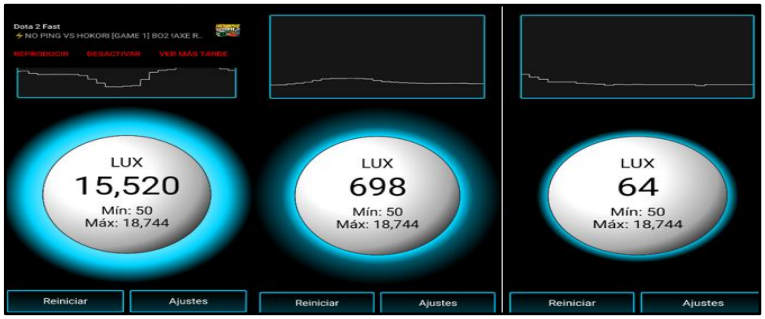


Figura 11-3: Cantidad de Lux

Realizado por: Guillen M., Bonilla T., 2021

Y para la distancia se obtuvo el criterio de igualmente 3 rangos:

- Rango mayor a 3 m
- Rango 2 entre 2 a 3 m
- Rango 1 entre 1 a 1 m

Por ello se realizó las pruebas con estos dos criterios tomando en cuenta la distancia y el número luxes. Se realizaron 3 pruebas por cada rango de iluminación cada prueba con 3 distancia diferentes de acuerdo con los rangos de distancia antes mencionados.

- Tablas de iluminación entre 5000 y 15000 lux

Tabla 1-3: Detección entre 5000 y 1500 lux a más de 3 metros de distancia

N°	Objeto	N° de pruebas	N° de pruebas detectados	N° de pruebas no detectados
1	Personas	4	4	0
2	Bicicletas	4	4	0
3	Automóviles	4	4	0
4	Buses	4	4	0
5	Gatos	4	0	4
6	Perros	4	0	4
7	Mesas	4	4	0
8	Sillas	4	3	1
9	Televisiones	4	1	3
10	Cuchillos	4	0	4
11	Laptops	4	1	3
12	Celulares	4	0	4
13	Libros	4	0	4
14	Refrigeradoras	4	3	1
15	Microondas	4	0	4
16	Pelotas	4	0	4
17	Paraguas	4	0	4
18	Baños (retretes)	4	0	4
19	Camas	4	3	1
20	Frutas, bananas, manzanas	4	0	4
Total		80	31	49
Eficiencia		38,75		

Realizado por: Guillen M., Bonilla T., 2021

Tabla 2-3: Detección entre 5000 y 1500 lux entre 2 y 3 metros de distancia

N°	Objeto	N° de pruebas	N° de pruebas detectados	N° de pruebas no detectados
1	Personas	4	4	0
2	Bicicletas	4	4	0
3	Automóviles	4	4	0
4	Buses	4	4	0
5	Gatos	4	2	2
6	Perros	4	2	2
7	Mesas	4	4	0
8	Sillas	4	4	0
9	Televisiones	4	4	0
10	Cuchillos	4	1	3
11	Laptops	4	1	3
12	Celulares	4	0	4
13	Libros	4	1	3
14	Refrigeradoras	4	4	0
15	Microondas	4	2	2
16	Pelotas	4	1	3
17	Paraguas	4	1	3
18	Baños (retretes)	4	3	1
19	Camas	4	4	0
20	Frutas, bananas, manzanas	4	0	4
Total		80	50	30
Eficiencia		62,5		

Realizado por: Guillen M., Bonilla T., 2021

Tabla 3-3: Detección entre 5000 y 1500 lux entre 1 y 2 metros de distancia

N°	Objeto	N° de pruebas	N° de pruebas detectados	N° de pruebas no detectados
1	Personas	4	4	0
2	Bicicletas	4	4	0
3	Automóviles	4	1	3
4	Buses	4	0	4
5	Gatos	4	4	0
6	Perros	4	4	0
7	Mesas	4	4	0
8	Sillas	4	4	0
9	Televisiones	4	4	0
10	Cuchillos	4	3	1
11	Laptops	4	3	1
12	Celulares	4	4	0
13	Libros	4	4	0
14	Refrigeradoras	4	3	1
15	Microondas	4	2	2
16	Pelotas	4	3	1
17	Paraguas	4	3	1
18	Baños (retretes)	4	3	1
19	Camas	4	4	0
20	Frutas, bananas, manzanas	4	2	2
Total		80	63	17
Eficiencia		78,75		

Realizado por: Guillen M., Bonilla T., 2021

- Tablas de iluminación entre 1500y 500 lux

Tabla 4-3: Detección entre 1500 y 500 lux a más de 3 metros de distancia

N°	Objeto	N° de pruebas	N° de pruebas detectados	N° de pruebas no detectados
1	Personas	4	4	0
2	Bicicletas	4	4	0
3	Automóviles	4	4	0
4	Buses	4	4	0
5	Gatos	4	0	4
6	Perros	4	0	4
7	Mesas	4	2	2
8	Sillas	4	3	1
9	Televisiones	4	0	4
10	Cuchillos	4	0	4
11	Laptops	4	0	4
12	Celulares	4	0	4
13	Libros	4	0	4
14	Refrigeradoras	4	3	1
15	Microondas	4	2	2
16	Pelotas	4	0	4
17	Paraguas	4	0	4
18	Baños (retretes)	4	3	1
19	Camas	4	3	1
20	Frutas, bananas, manzanas	4	0	4
Total		80	32	48
Eficiencia		40		

Realizado por: Guillen M., Bonilla T., 2021

Tabla 5-3: Detección entre 1500 y 500 lux entre 2 y 3 metros de distancia

N°	Objeto	N° de pruebas	N° de pruebas detectados	N° de pruebas no detectados
1	Personas	4	4	0
2	Bicicletas	4	4	0
3	Automóviles	4	4	0
4	Buses	4	4	0
5	Gatos	4	1	3
6	Perros	4	3	1
7	Mesas	4	4	0
8	Sillas	4	4	0
9	Televisiones	4	0	4
10	Cuchillos	4	0	4
11	Laptops	4	3	1
12	Celulares	4	0	4
13	Libros	4	1	3
14	Refrigeradoras	4	4	0
15	Microondas	4	2	2
16	Pelotas	4	1	3
17	Paraguas	4	0	4
18	Baños (retretes)	4	3	1
19	Camas	4	3	1
20	Frutas, bananas, manzanas	4	0	4
Total		80	45	35
Eficiencia		40		

Realizado por: Guillen M., Bonilla T., 2021

Tabla 6-3: Detección entre 1500 y 500 lux entre 1 y 2 metros de distancia

N°	Objeto	N° de pruebas	N° de pruebas detectados	N° de pruebas no detectados
1	Personas	4	4	0
2	Bicicletas	4	4	0
3	Automóviles	4	2	2
4	Buses	4	1	3
5	Gatos	4	4	0
6	Perros	4	4	0
7	Mesas	4	4	0
8	Sillas	4	4	0
9	Televisiones	4	4	0
10	Cuchillos	4	4	0
11	Laptops	4	4	0
12	Celulares	4	4	0
13	Libros	4	4	0
14	Refrigeradoras	4	3	1
15	Microondas	4	4	0
16	Pelotas	4	3	1
17	Paraguas	4	3	1
18	Baños (retretes)	4	4	0
19	Camas	4	4	0
20	Frutas, bananas, manzanas	4	2	2
Total		80	70	10
Eficiencia		87,5		

Realizado por: Guillen M., Bonilla T., 2021

- Tablas de iluminación entre 400 y 50 lux

Tabla 7-3: Detección entre 400 y 50 lux a más de 3 metros de distancia

N°	Objeto	N° de pruebas	N° de pruebas detectados	N° de pruebas no detectados
1	Personas	4	4	0
2	Bicicletas	4	4	0
3	Automóviles	4	2	2
4	Buses	4	3	1
5	Gatos	4	0	4
6	Perros	4	0	4
7	Mesas	4	3	1
8	Sillas	4	1	3
9	Televisiones	4	0	4
10	Cuchillos	4	0	4
11	Laptops	4	0	4
12	Celulares	4	0	4
13	Libros	4	0	4
14	Refrigeradoras	4	4	0
15	Microondas	4	1	3
16	Pelotas	4	0	4
17	Paraguas	4	0	4
18	Baños (retretes)	4	0	4
19	Camas	4	3	1
20	Frutas, bananas, manzanas	4	0	4
Total		80	25	55
Eficiencia		31,25		

Realizado por: Guillen M., Bonilla T., 2021

Tabla 8-3: Detección entre 400 y 50 lux entre 2 y 3 metros de distancia

N°	Objeto	N° de pruebas	N° de pruebas detectados	N° de pruebas no detectados
1	Personas	4	4	0
2	Bicicletas	4	2	2
3	Automóviles	4	3	1
4	Buses	4	3	1
5	Gatos	4	4	0
6	Perros	4	4	0
7	Mesas	4	3	1
8	Sillas	4	4	0
9	Televisiones	4	3	1
10	Cuchillos	4	1	3
11	Laptops	4	2	2
12	Celulares	4	0	4
13	Libros	4	0	4
14	Refrigeradoras	4	3	1
15	Microondas	4	1	3
16	Pelotas	4	0	4
17	Paraguas	4	0	4
18	Baños (retretes)	4	4	0
19	Camas	4	4	0
20	Frutas, bananas, manzanas	4	0	4
Total		80	45	35
Eficiencia		56,25		

Realizado por: Guillen M., Bonilla T., 2021

Tabla 9-3: Detección entre 400 y 50 lux entre 1 y 2 metros de distancia

N°	Objeto	N° de pruebas	N° de pruebas detectados	N° de pruebas no detectados
1	Personas	4	4	0
2	Bicicletas	4	4	0
3	Automóviles	4	1	3
4	Buses	4	0	4
5	Gatos	4	4	0
6	Perros	4	4	0
7	Mesas	4	4	0
8	Sillas	4	4	0
9	Televisiones	4	3	1
10	Cuchillos	4	2	2
11	Laptops	4	4	0
12	Celulares	4	4	0
13	Libros	4	3	1
14	Refrigeradoras	4	3	1
15	Microondas	4	4	0
16	Pelotas	4	4	0
17	Paraguas	4	3	1
18	Baños (retretes)	4	4	0
19	Camas	4	4	0
20	Frutas, bananas, manzanas	4	4	0
Total		80	68	10
Eficiencia		85		

Realizado por: Guillen M., Bonilla T., 2021

3.3 Análisis de tiempo de respuesta del sistema

Pruebas tiempo de respuesta para ellos se integró un comando en la programación de Python el cual calcula el tiempo desde que lee la imagen hasta el tiempo que es reproducida como audio y variando el número de objetos en pantalla, se muestra estos resultados en la tabla 10-3.

Tabla 10-3: Tiempo de respuesta

N°	N° de objetos en cada prueba	N° de pruebas	Tiempo de respuesta promedio
1	1	4	0,4231
2	2	4	0,50129
3	3	4	0,53712
4	4	4	0,5732
5	5	4	0,61489

Realizado por: Guillen M., Bonilla T., 2021

3.4 Análisis de confiabilidad del sistema

Se usó como técnica de recolección de datos la observación experimental ya que se realizaron pruebas en base a la experimentación final del sistema para esto se emplea como técnica de recolección de datos una hoja de registro cuando el sistema detecta la imagen. Se determinó una muestra de elementos mediante la siguiente fórmula estadística.

$$n = \frac{Z^2 * p * q * N}{e^2 * (N-1) + Z^2 * p * q}$$

Donde:

- **n:** tamaño de la muestra
- **N:** tamaño de la población o universo
- **Z:** constante que depende del nivel de confianza que se asigne p: probabilidad de éxito
- **q:** probabilidad de fracaso
- **e:** error muestral deseado en tanto por ciento

Para determinar el valor del nivel de confianza tenemos la siguiente tabla de datos, esta tabla fue creada por Carolina Jara y Carlos Caba quienes también realizaron pruebas a su prototipo que efectúa reconocimiento de imágenes, cuya tabla de nivel de confianza sirvió de referencia para las pruebas del prototipo. Esto se muestra en la tabla 11-3.

Tabla 11-3: Tiempo de respuesta

Valor Z	1,645	1,96	2,24	2,576
Nivel de confianza	90%	95%	97,5%	99%

Realizado por: Guillen M., Bonilla T., 2021

Para el sistema se consideró los intentos que se hicieron para la detección de las señales, por lo que es necesario establecer un nivel de confianza del 87% considerándolo como aceptable del cual se obtiene un valor de Z de 1,96 el cual fue usado para el cálculo de la muestra. El error que se puede aceptar en el modelo es del 13% ya se le asigna un nivel de confianza al sistema del 87%

El siguiente paso es recolectar la información necesaria para la muestra calculada para lo cual se considera como aceptable el reconocimiento hasta dos intentos y el tercero se considera como no aceptable.

Mediante el análisis experimental realizado al sistema se llegó a la conclusión que existe un 87% de aciertos en el reconocimiento que se considera aceptable para los sistemas de reconocimiento existentes. Se puede observar en el gráfico 1-3 el resultado obtenido del análisis del sistema.

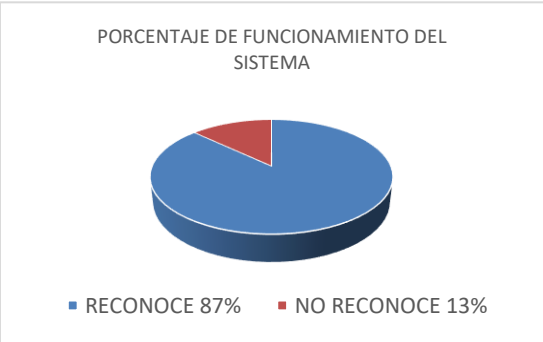


Gráfico 1-3: Porcentaje de funcionamiento del sistema

Realizado por: Guillen M., Bonilla T., 2021

CAPÍTULO IV

4. EVALUACIÓN ECONÓMICA

Para la construcción e implementación del prototipo se hizo uso de diferentes equipos y materiales indispensables, dichos elementos se detallan en los siguientes apartados.

4.1 Costo de *Hardware*

Dentro de los costos directos de fabricación tenemos todos aquellos que forman parte de la construcción e implementación del prototipo electrónico, todos los valores correspondientes a cada parte del sistema se informan en la Tabla 1-4.

Tabla 1-4: Costos directos de fabricación del prototipo

Cantidad	Componentes	Costo unitario	Costo total
1	Raspberry Pi 4	$150	$185
1	Cámara de 8Mpx Cámara Pi NoIR V2	$80	$80
1	Batería Li-po	$45	$45
1	Micro SD	$25	$25
1	Parlantes	$50	$50
1	Step Down 5ª XL4015	$15	$15
1	Cables varios	$100	$100
1	Impresiones 3D	$80	$80
1	Ventilador	$40	$40
1	Cargador de batería	$60	$60
1	Computador	500	500
TOTAL			$1145

Realizado por: Guillen M., Bonilla T., 2021.

4.2 Costos de *Software*

El software del sistema implementado se detalla en la tabla 2-4.

Tabla 2-4: Costos del *Software* del prototipo

Cantidad	Componentes	Costo unitario	Costo total
1	Python	$0	$0
1	Open CV	$0	$0
1	Raspbian Stretch	$0	$0
TOTAL			$0

Realizado por: Guillen M., Bonilla T., 2021.

4.3 Costo total de implementación

El costo total de implementación del programa se calcula en función de los costos anteriormente definidos recopilados en la tabla 1-4 y 2-4.

Tabla 3-4: Costo total del dispositivo IoT

Cantidad	Componentes	Costo unitario	Costo total
1	Costos Hardware	$1145	$1145
1	Costos Software	$0	$0
TOTAL			$1145

Realizado por: Guillen M., Bonilla T., 2021.

CONCLUSIONES

- El prototipo realizado permitió implementar un dispositivo electrónico de ayuda a personas con deficiencia visual como asistente para poder interactuar dentro de lugares indoor, brindando alta fiabilidad al sistema.

- Al utilizar técnicas de Visión Artificial, se construyó un prototipo electrónico para poder ayudar a personas con deficiencia visual para lo cual se utilizaron métodos y técnicas de pre-procesamiento y procesamiento de imágenes que permitieron comparar la imagen capturada en tiempo real, con una serie de imágenes indexadas al programa para que mediante un mensaje auditivo dar a conocer al usuario del sistema el objeto detectado y a que posición del individuo se encuentra.

- Las pruebas realizadas al prototipo para el reconocimiento en lugares indoor, dieron valores porcentuales altos en su mayoría dando como resultado un valor porcentual total de 87% dando una acorde aprobación al algoritmo.

- El funcionamiento del prototipo implementado permite reducir un 20% de accidentes que existe en la actualidad, mediante la visión artificial y los algoritmos de detección se crea sistemas que ayudan con la supervisión y seguridad de los individuos en todos los entornos que se encuentren.

- Al seleccionar los elementos para la implementación del sistema se busca reducir costos, pero sin perder la calidad es decir se seleccionan los óptimos, para la creación del algoritmo de detección se emplea el HAAR Cascade porque ocupa menos recursos al momento de iniciar el entrenamiento.

RECOMENDACIONES

- Realizar una retroalimentación de todas las fases, componentes y algoritmos que se van a implementar para que no muestre fallas en algún punto del desarrollo o en la fase final de la creación del sistema de seguridad.

- Determinar y establecer todos los componentes a utilizar en el desarrollo del prototipo, para el desarrollo del software determinar el conjunto de imágenes positivas y negativas que se vayan almacenando en una base de datos para facilitar y reducir el tiempo de entrenamiento.

- Realizar varias pruebas para evaluar el funcionamiento del sistema, para que el tiempo de entrenamiento sea mínimo se debe emplear una computadora que posea características mínimas como memoria RAM de y procesador.

- Realizar estudios para crear un dispositivo que funciones bajo condiciones adversas como la lluvia, niveles de temperatura elevada, baja luminosidad y que el prototipo funcione en la noche si detectar falsos positivos.

BIBLIOGRAFÍA

DI GENNARO, Elena. Sistema de Detección de Obstáculos para Invidentes. *Urbe.* [En línea] 2017. [Citado el: 02 de Noviembre de 2021.] https://virtual.urbe.edu/tesispub/0105666/intro.pdf.

ASKIX. Dispositivo de retroalimentación háptica para invidentes. *Askix.* [En línea] 2020. [Citado el: 11 de Julio de 2021.] https://www.askix.com/dispositivo-de-retroalimentacion-haptica-para-invidentes-proyecto-halo.html.

AULACLIC. Espectro electromagnético. *Aulaclic.* [En línea] 2020. [Citado el: 18 de Julio de 2021.] https://www.aulaclic.es/fotografia-photoshop/graficos/espectro1.gif.

CONADIS. Estadísticas de Discapacidad. *Consejo Nacional para la Igualdad de Discapacidades.* [En línea] 2021a. [Citado el: 01 de Noviembre de 2021.] https://www.consejodiscapacidades.gob.ec/estadisticas-de-discapacidad/.

CONADIS. Personas con discapacidad por provincia. *Consejo Nacional para la Igualdad de Discapacitados.* [En línea] 2016b. [Citado el: 04 de Noviembre de 2021.] https://www.consejodiscapacidades.gob.ec/wp-content/uploads/downloads/2015/09/estadistica_conadis.pdf.

DUARTE DUARTE, Jakeline. Ambientes de aprendizaje una aproximación conceptual. *Rieoei.* [En línea] 2018. [Citado el: 14 de Agosto de 2021.] https://rieoei.org/RIE/article/download/2961/3875/.

DUNAI, Larisa; et al. Dispositivo de navegación para personas invidentes basado en la tecnología Time of Light. *Scielo.* [En línea] 2013. [Citado el: 16 de Julio de 2021.] http://www.scielo.org.co/pdf/dyna/v80n179/v80n179a04.pdf.

ECURED. Raspbian. *EcuRed.* [En línea] 2020. [Citado el: 16 de Agosto de 2021.] https://www.ecured.cu/Raspbian.

EPA. LCD´s Gráficos. *Diario electronico hoy.* [En línea] 2018. [Citado el: 18 de Julio de 2021.] https://www.diarioelectronicohoy.com/blog/lcds-graficos-pcd8544.

GRUPOELECTROSTORE. Tarjetas de memoria SD Sandisk 32 GB. *Grupoelectrostore.* [En línea] 2019. [Citado el: 10 de Junio de 2021.] https://grupoelectrostore.com/shop/tarjetas-de-memoria/tarjeta-de-memoria-sd-sandisk-32gb-clase-10-original/.

INFOSALUS. La OMS estima que hay 285 millones de personas con discapacidad visual en el mundo. *Infosalus.* [En línea] 2016. [Citado el: 10 de Abril de 2021.] https://www.infosalus.com/actualidad/noticia-oms-estima-hay-285-millones-personas-discapacidad-visual-mundo-20131010134206.html.

KHAN, Nayla. HALO. *Walyou*. [En línea] 2010. [Citado el: 11 de Julio de 2021.] https://walyou.com/blog/2010/12/17/haptic-assistance-for-the-blind/.

LAZZUS. Lazzus. *Lazzus.* [En línea] 2021. [Citado el: 10 de Julio de 2021.] http://lazzus.com/es/.

LUDEÑA, Juan. Implementación de un sistema de seguridad para supervisión de niños entre 2 a 4 años usando visión artificial. *Espoch*. [En línea] 2019. [Citado el: 14 de Julio de 2021.] http://dspace.espoch.edu.ec/bitstream/123456789/13593/1/108T0299.pdf.

MARÍN, Rafael. OpenCv. *Revistadigital.* [En línea] 2020. [Citado el: 11 de Julio de 2021.] https://revistadigital.inesem.es/informatica-y-tics/opencv/.

MATI. Baja visión y la discapacidad visual. *Mati.* [En línea] 2020. [Citado el: 12 de Agosto de 2021.] http://www.webmati.es/index.php?option=com_content&view=article&id=23:baja-vision-y-la-discapacidad-visual&catid=13.

MGSYSTEM. Batería Zippy 7.4 V. *Mgsystem.* [En línea] 2021. [Citado el: 19 de Junio de 2021.] https://articulo.mercadolibre.com.ec/MEC-503039587-mgsystem-lipo-bateria-zippy-1000mah-2s-74v-robot-25c-_JM#position=1&search_layout=stack&type=item&tracking_id=92be6ebd-7ca0-405a-b8d8-a163370622e5.

MINISTERIO DEL DESARROLLO SOCIAL. Orientación y Movilidad: estrategias frente a nuevos escenarios poblacionales. *Pronadis.* [En línea] 2013. [Citado el: 11 de Agosto de 2021.] https://pronadis.mides.gub.uy/23113/orientacion-y-movilidad:-estrategias-frente-a--nuevos-escenarios-poblacionales.

MURILLO, Otto; & SERNA, Carlos. Prototipo de baatón inteligente para personas con limitación. *Ucp*. [En línea] 2017. [Citado el: 15 de Agosto de 2021.] https://repositorio.ucp.edu.co/bitstream/10785/4634/4/DDMIST18.pdf.

OPENWEBINAR. Phyton. *Openwebinar*. [En línea] 2019. [Citado el: 18 de Julio de 2021.] http://theganSgtersnsk.blogspot.com/2015/03/lenguaje-de-programacion-python.html.

ORGANIZACIÓN MUNDIAL DE LA SALUD. Informe mundial sobre la visión. *OMS.* [En línea] 2020. [Citado el: 13 de Agosto de 2021.] https://apps.who.int/iris/bitstream/handle/10665/331423/9789240000346-spa.pdf.

ORGANIZACIÓN NACIONAL DE CIEGOS ESPAÑOLES. Discapacidad visual y autonomía personal. *Sid.* [En línea] 2011. [Citado el: 16 de Agosto de 2021.] https://sid.usal.es/idocs/F8/FDO26230/discap_visual.pdf.

PAJARES, Gonzalo; & CRUZ, Jesús. *Visión por computador, imágenes digitales y aplicaciones.* Madrid : RA-MA, 2001.

PÉREZ, David. Sistemas embebidos y Sistemas operatios embebidos. *Ucv.* [En línea] 2009. [Citado el: 10 de Julio de 2021.] https://www.academia.edu/16523506/Info_Sistemas_Embebidos.

PROGRAMADOR CLIC. Análisis de extracción de características SIFT. *Programador clic.* [En línea] 2018. [Citado el: 10 de Julio de 2021.] https://programmerclick.com/article/4389154959/.

RASPBERRY PI. Aprende Raspberry PI. *Raspberry PI.* [En línea] 2018a. [Citado el: 21 de Julio de 2021.] https://www.raspberrypi.org/..

RASPEBERRY PI. Módulo de cámara Raspberry Pi 2 NoIR. *Raspberry.* [En línea] 2016b. [Citado el: 19 de Agosto de 2021.] https://www.raspberrypi.com/products/pi-noir-camera-v2/.

RASPBERRY PI. Descargas Raspberry Pi. *Raspberrypi.* [En línea] 2016c. [Citado el: 01 de Septiembre de 2021.] https://www.raspberrypi.com/software/.

REVUELTA SANZ, Pablo. Nuevo prototipo sónico de ayuda a la ceguera. *Dicyt.* [En línea] 2013. [Citado el: 15 de Julio de 2021.] https://www.dicyt.com/viewNews.php?newsId=29798.

SALGADO, Luis. Visión artificial: Fundamentos y aplicaciones. *Arantxa.* [En línea] 2007. [Citado el: 10 de Junio de 2021.] http://arantxa.ii.uam.es/~jms/seminarios_doctorado/abstracts2006-2007/20070503LSalagado.pdf.

SEELIGHT. Seelight. *Seelight.* [En línea] 2020. [Citado el: 16 de Julio de 2021.] https://seelight.site/.

SEGURA MEDRANDA, Oscar Daniel. Diseño y construcción de un sistema eectrónico para personas no videntes como ayuda para el cruce de las calles urbanas basado en el procesamiento

de imágenes. *Espoch.* [En línea] 2019. [Citado el: 03 de Noviembre de 2021.] http://dspace.espoch.edu.ec/bitstream/123456789/13508/1/108T0290.pdf.

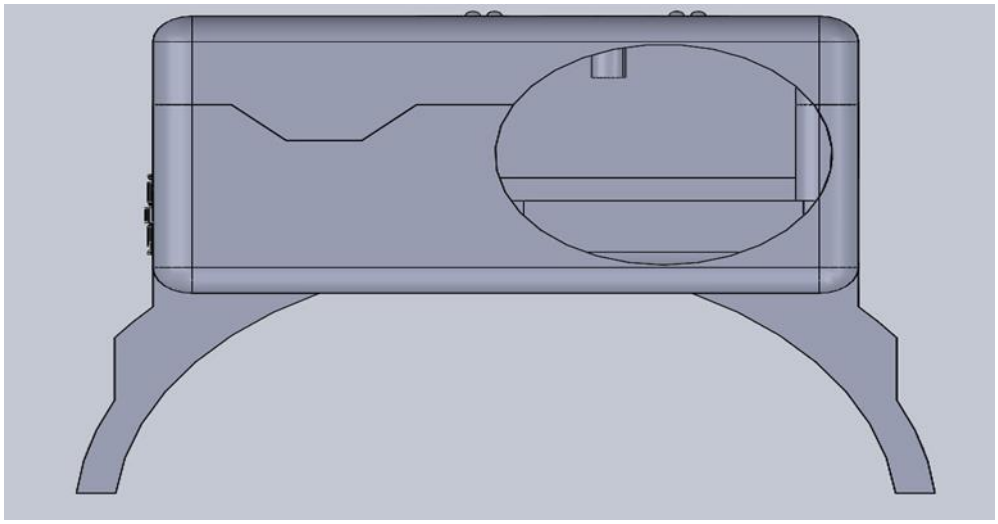
UNIT ELECTRONICS. Modulo de carga Step Down XL4015 5A. *Uelectronics.* [En línea] 2020. [Citado el: 12 de Junio de 2021.] https://uelectronics.com/producto/modulo-de-carga-reductor-xl4015-5a-con-medidores/.

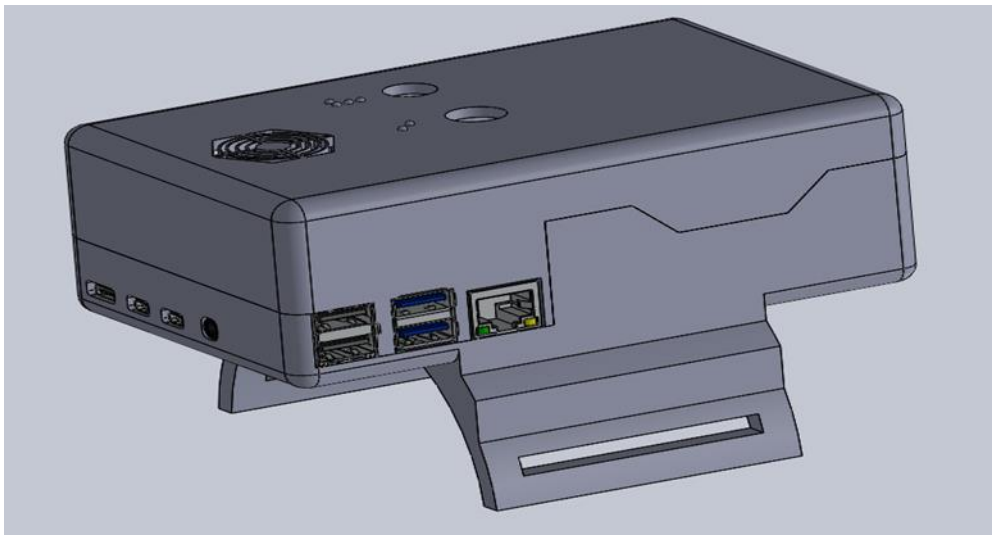
VÉLEZ, José. Detección de contornos. *Vision por computador.* [En línea] 2007. [Citado el: 14 de Julio de 2021.] http://www.visionporcomputador.es/libroVision/VisionPorComputador.pdf.

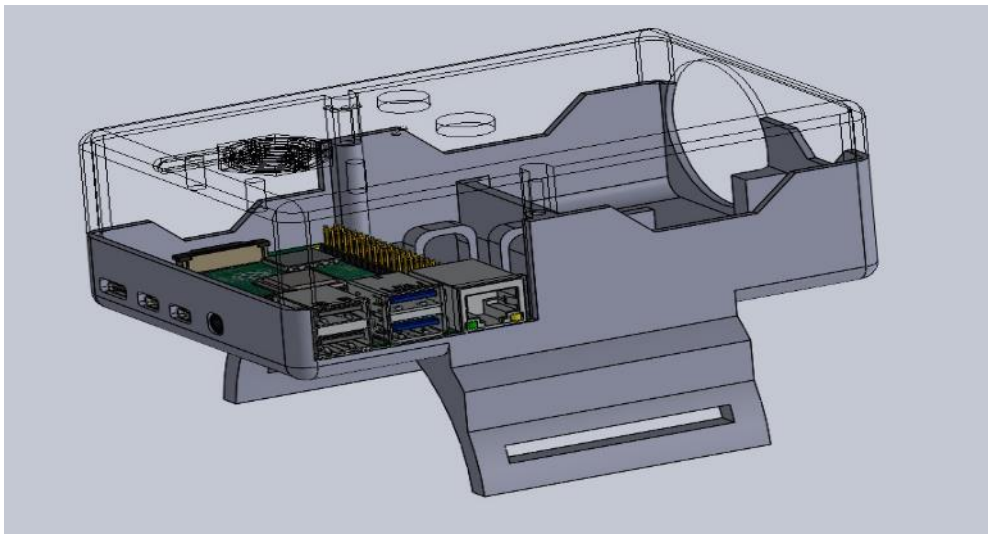
XATAKANDROID. Aplicaión móvil. *Xatakandroid.* [En línea] 2020. [Citado el: 13 de Julio de 2021.] https://www.xatakandroid.com/.

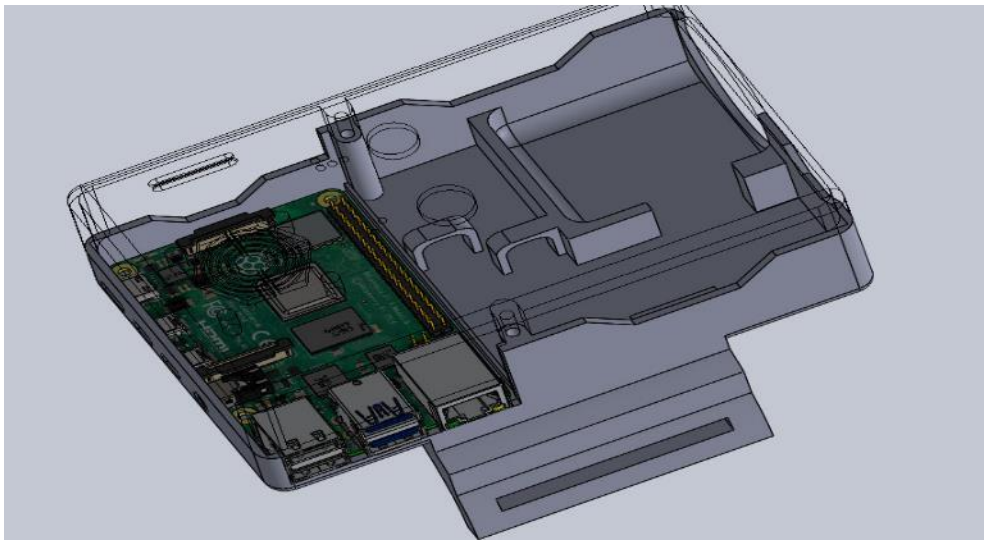
ANEXOS

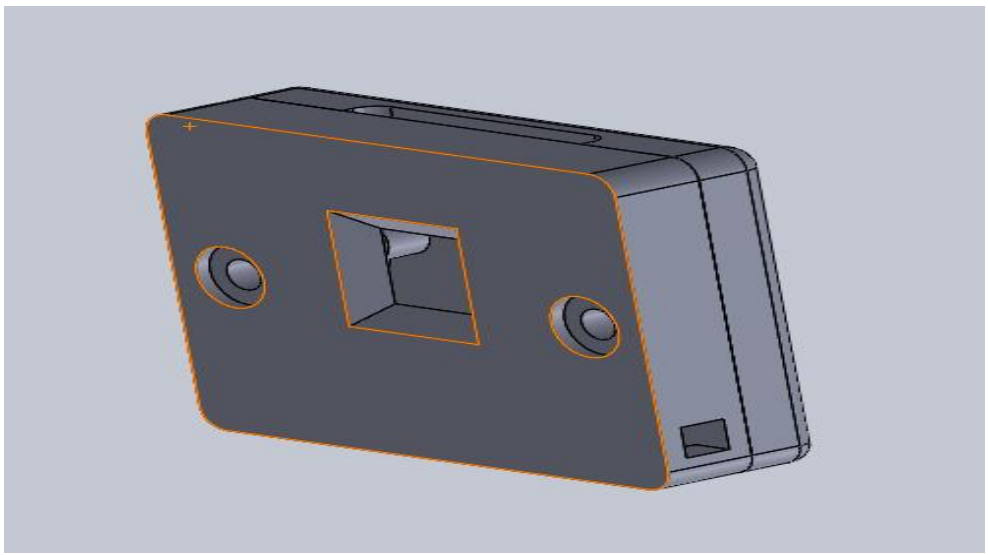
ANEXO A: CREACIÓN DEL PROTOTIPO EN SOLIDWORKS



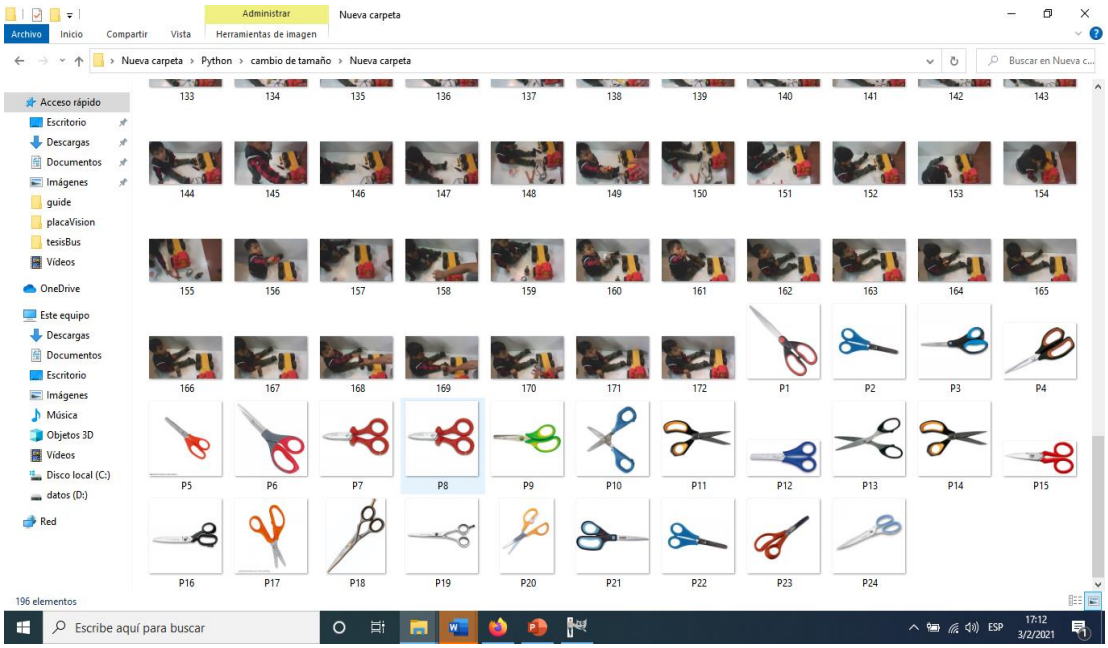




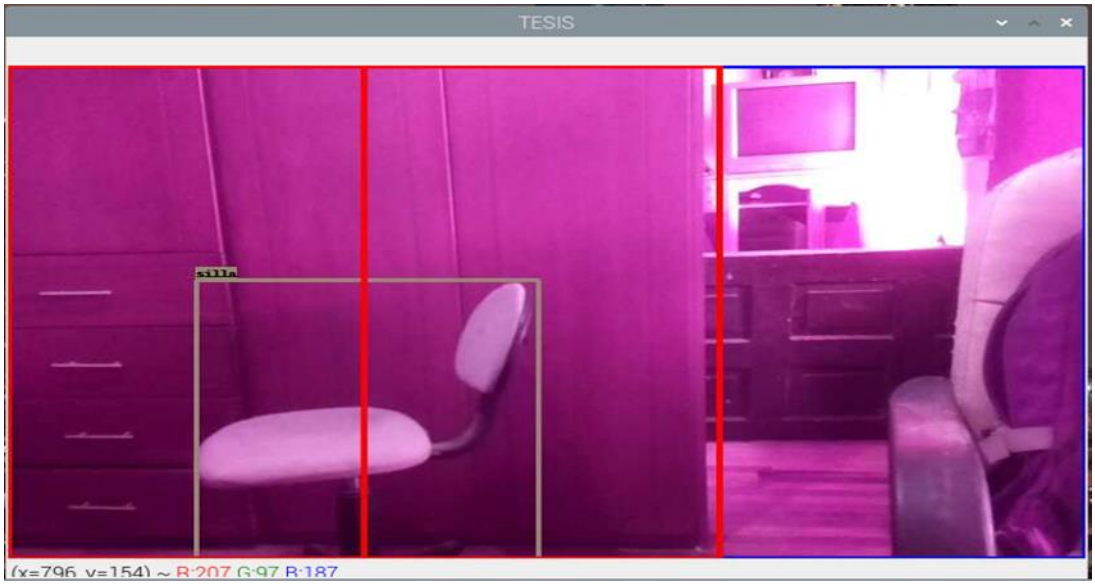


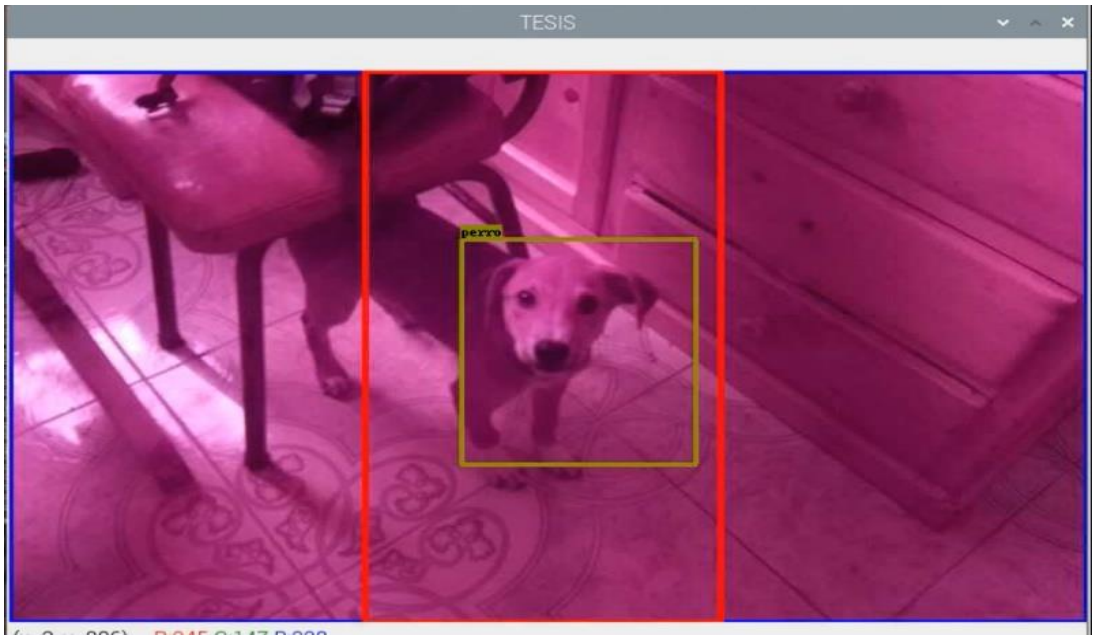


ANEXO B: TOMA DE MUESTRAS

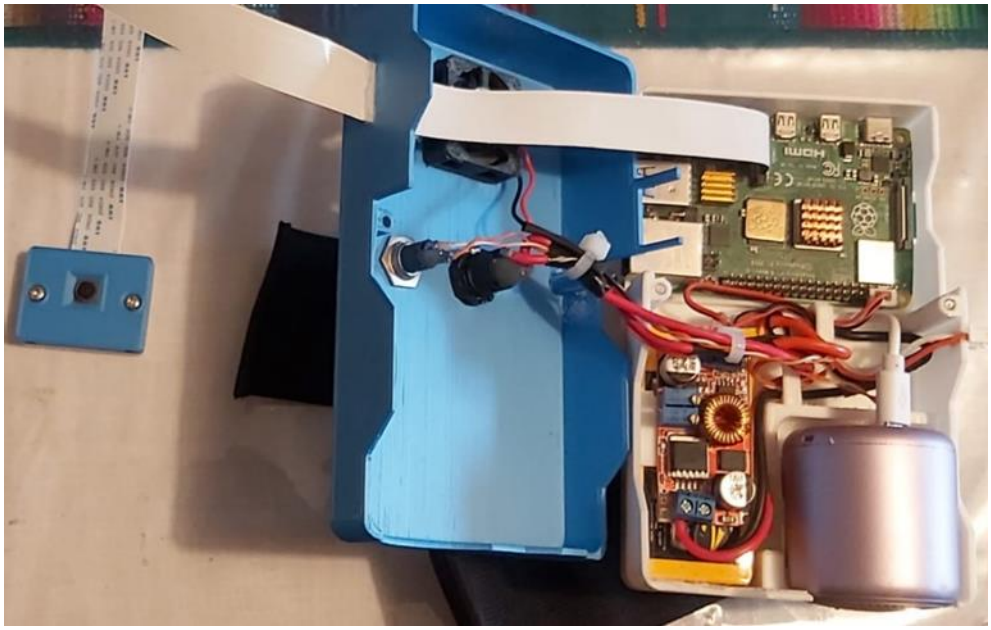


ANEXO C: PRUEBAS DEL ALGORITMO





ANEXO D: ESQUEMA ELECTRÓNICO



ANEXO E: PRUEBA FINAL DEL PROTOTIPO

ANEXO F: PROGRAMACIÓN

```
import os
import cv2
import math
import numpy as np
from picamera.array import PiRGBArray
from picamera import PiCamera
from gtts import gTTS
import random

personaClassif = cv2.CascadeClassifier('personasCascade.xml')
biciClassif = cv2.CascadeClassifier('bicicletasCascade.xml')
autoClassif = cv2.CascadeClassifier('autoCascade.xml')
busesClassif = cv2.CascadeClassifier('busesCascade.xml')
gatosClassif = cv2.CascadeClassifier('gatosCascade.xml')
perrosClassif = cv2.CascadeClassifier('perrosCascade.xml')
mesasClassif = cv2.CascadeClassifier('mesasCascade.xml')
sillasClassif = cv2.CascadeClassifier('sillasCascade.xml')
tvClassif = cv2.CascadeClassifier('tvCascade.xml')
cuchillosClassif = cv2.CascadeClassifier('cuchillosCascade.xml')
laptopClassif = cv2.CascadeClassifier('laptopCascade.xml')
celularesClassif = cv2.CascadeClassifier('celuCascade.xml')
librosClassif = cv2.CascadeClassifier('librosCascade.xml')
refriClassif = cv2.CascadeClassifier('refriCascade.xml')
microClassif = cv2.CascadeClassifier('microCascade.xml')
pelotasClassif = cv2.CascadeClassifier('pelotasCascade.xml')
paraguasClassif = cv2.CascadeClassifier('paraguasCascade.xml')
banoClassif = cv2.CascadeClassifier('banoCascade.xml')
camaClassif = cv2.CascadeClassifier('camaCascade.xml')
frutasClassif = cv2.CascadeClassifier('frutasCascade.xml')
camara = PiCamera()
camara.resolution = (800,480)
camara.framerate = 10
imagen = PiRGBArray(camera, size=(800,480))
imagen.truncate(0)
Activar=True
activador=False
valorI=1
valorF=2
pos1='esta en la arriba en la izquierda'
pos2='esta en la abajo en la izquierda'
pos3='esta en al frente'
pos5='esta abajo'
pos6='esta en la arriba en la derecha'
pos7='esta en la abajo en la derecha'
colores = [
    'AliceBlue', 'Chartreuse', 'Aqua', 'Aquamarine', 'Azure', 'Beige', 'Bisque',
    'BlanchedAlmond', 'BlueViolet', 'BurlyWood', 'CadetBlue', 'AntiqueWhite',
    'Chocolate', 'Coral', 'CornflowerBlue', 'Cornsilk', 'Crimson', 'Cyan',
    'DarkCyan', 'DarkGoldenRod', 'DarkGrey', 'DarkKhaki', 'DarkOrange',
    'DarkOrchid', 'DarkSalmon', 'DarkSeaGreen', 'DarkTurquoise', 'DarkViolet',
    'DeepPink', 'DeepSkyBlue', 'DodgerBlue', 'FireBrick', 'FloralWhite',
    'ForestGreen', 'Fuchsia', 'Gainsboro', 'GhostWhite', 'Gold', 'GoldenRod',
    'Salmon', 'Tan', 'HoneyDew', 'HotPink', 'IndianRed', 'Ivory', 'Khaki',
    'Lavender', 'LavenderBlush', 'LawnGreen', 'LemonChiffon', 'LightBlue',
    'LightCoral', 'LightCyan', 'LightGoldenRodYellow', 'LightGray', 'LightGrey',
    'LightGreen', 'LightPink', 'LightSalmon', 'LightSeaGreen', 'LightSkyBlue',
    'LightSlateGray', 'LightSlateGrey', 'LightSteelBlue', 'LightYellow', 'Lime',
```

```
    'LimeGreen', 'Linen', 'Magenta', 'MediumAquaMarine', 'MediumOrchid',
    'MediumPurple', 'MediumSeaGreen', 'MediumSlateBlue', 'MediumSpringGreen',
    'MediumTurquoise', 'MediumVioletRed', 'MintCream', 'MistyRose', 'Moccasin',
    'NavajoWhite', 'OldLace', 'Olive', 'OliveDrab', 'Orange', 'OrangeRed',
    'Orchid', 'PaleGoldenRod', 'PaleGreen', 'PaleTurquoise', 'PaleVioletRed',
    'PapayaWhip', 'PeachPuff', 'Peru', 'Pink', 'Plum', 'PowderBlue', 'Purple',
    'Red', 'RosyBrown', 'RoyalBlue', 'SaddleBrown', 'Green', 'SandyBrown',
    'SeaGreen', 'SeaShell', 'Sienna', 'Silver', 'SkyBlue', 'SlateBlue',
    'SlateGray', 'SlateGrey', 'Snow', 'SpringGreen', 'SteelBlue', 'GreenYellow',
    'Teal', 'Thistle', 'Tomato', 'Turquoise', 'Violet', 'Wheat', 'White',
    'WhiteSmoke', 'Yellow', 'YellowGreen'
]

Objetos = [
    'persona', 'bicicleta', 'Auto', 'buses', 'gatos', 'perros', 'mesas',
    'sillas', 'tv', 'cuchillo', 'laptop', 'celulares',
    'libros', 'refrigeradora', 'microondas', 'pelota', 'paraguas', 'baño',
    'cama','frutas'
]
texto=""

frame=camara.capture_continuous(imagen, format="bgr")
ancho = int(frame.get(cv2.CAP_PROP_FRAME_WIDTH))
alto = int(frame.get(cv2.CAP_PROP_FRAME_HEIGHT))
a = math.ceil(ancho * (0))
b = math.ceil(ancho * (0.33))
c = math.ceil(ancho * (0.66))
d = math.ceil(ancho * (1))
A = math.ceil(alto * (0))
A = math.ceil(alto * (0.5))
A = math.ceil(alto * (1))

while Activar:
    frame=camara.capture_continuous(imagen, format="bgr")
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    if(valorI==1):
        personas = personasClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors =
91,minSize=(70,78))
    if(valorI==2):
        bici = biciClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==3):
        auto = autoClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==4):
        buses = busesClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==5):
        gatos = gatosClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==6):
        perros = perrosClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==7):
        mesas = mesasClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==8):
        sillas = sillasClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==9):
        tv = tvClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==10):
        personas = personasClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors =
91,minSize=(70,78))
    if(valorI==11):
        cuchillos = cuchillosClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors =
91,minSize=(70,78))
```

```
    if(valorI==12):
        celu = celularesClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==13):
        libros = librosClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==14):
        refri = refriClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==15):
        micro = microClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==16):
        pelotas = pelotasClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==17):
        paraguas = paraguasClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors =
91,minSize=(70,78))
    if(valorI==18):
        bano = banoClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==19):
        camas = camaClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))
    if(valorI==20):
        frutas = frutasClassif.detectMultiScale(gray,scaleFactor = 5,minNeighbors = 91,minSize=(70,78))

    if(valorI==valorF):
        text=''
        detectar_obj()
        valorI=1
        if activador:
            reproducir()
            activador=False


  if cv2.waitKey(1) & 0xFF == ord('q'):
    break
cap.release()
cv2.destroyAllWindows()


    def detectar_obj():

        for (x,y,w,h) in personas:
            color= colores[round(random()*len(colores))]
            cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
            cv2.putText(frame,objetos[1],(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
            cv2.rectangle(frame, (x, y-10), (x + 20, y + 10), color, -1)
            xc =(x+w)/2
            yc =(y+h)/2
            comparar_posicion(xc,yc,1)

        for (x,y,w,h) in bici:
            color= colores[round(random()*len(colores))]
            cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
            cv2.putText(frame,objetos[2],(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
            cv2.rectangle(frame, (x, y-10), (x + 20, y + 10), color, -1)
            xc =(x+w)/2
            yc =(y+h)/2
            comparar_posicion(xc,yc,2)

        for (x,y,w,h) in auto:
            color= colores[round(random()*len(colores))]
            cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
            cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
            cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
```

```
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,3)

for (x,y,w,h) in buses:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,4)

for (x,y,w,h) in gatos:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,5)

for (x,y,w,h) in perros:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,6)

for (x,y,w,h) in mesas:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,7)

for (x,y,w,h) in sillas:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,8)
for (x,y,w,h) in tv:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,9)

 for (x,y,w,h) in cuchillos:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
```

```
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,10)

for (x,y,w,h) in laptop:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,11)

for (x,y,w,h) in celulares:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,12)

for (x,y,w,h) in libros:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,13)

for (x,y,w,h) in refri:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,14)

for (x,y,w,h) in micro:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,15)

for (x,y,w,h) in pelotas:
    color= colores[round(random()*len(colores))]
    cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
    cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
    cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
    xc =(x+w)/2
    yc =(y+h)/2
    comparar_posicion(xc,yc,16)
```

```
    for (x,y,w,h) in paraguas:
        color= colores[round(random()*len(colores))]
        cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
        cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
        cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
        xc =(x+w)/2
        yc =(y+h)/2
        comparar_posicion(xc,yc,17)

    for (x,y,w,h) in banos:
        color= colores[round(random()*len(colores))]
        cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
        cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
        cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
        xc =(x+w)/2
        yc =(y+h)/2
        comparar_posicion(xc,yc,18)

    for (x,y,w,h) in camas:
        color= colores[round(random()*len(colores))]
        cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
        cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
        cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
        xc =(x+w)/2
        yc =(y+h)/2
        comparar_posicion(xc,yc,19)

    for (x,y,w,h) in frutas:
        color= colores[round(random()*len(colores))]
        cv2.rectangle(frame, (x,y),(x+w,y+h),color,4)
        cv2.putText(frame,'personas',(x,y-10),2,0.7,(0,0,0),2,cv2.LINE_AA)
        cv2.rectangle(frame, (x, x), (x + w, y + h), color, -1)
        xc =(x+w)/2
        yc =(y+h)/2
        comparar_posicion(xc,yc,20)
cv2.imshow('frame',frame)
valorI=valorI+1
def comparar_posicion(xcentro, ycentro,numero):
    activador=True
    if(xcentro>a and xcentro<=b):
        if(ycentro>A and ycentro<B):
            texto=texto+objeto[numero]+pos1
        else:
            texto=texto+objeto[numero]+pos2

     if(xcentro>b and xcentro<=c):
        if(ycentro>A and ycentro<B):
            texto=texto+objeto[numero]+pos3
        else:
            texto=texto+objeto[numero]+pos4

     if(xcentro>c and xcentro<=d):
        if(ycentro>A and ycentro<B):
            texto=texto+objeto[numero]+pos5
        else:
            texto=texto+objeto[numero]+pos6
def reproducir():
    sonido = gTTS(text lang="es", slow=False)
    sonido.save("objeto.mp3")
    os.system("mpg321 objeto.mp3"
```

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

DIRECCIÓN DE BIBLIOTECAS Y RECURSOS DEL APRENDIZAJE

UNIDAD DE PROCESOS TÉCNICOS

REVISIÓN DE NORMAS TÉCNICAS, RESUMEN Y BIBLIOGRAFÍA

Fecha de entrega: 22 / 12 / 2021

INFORMACIÓN DE LOS AUTORES
Nombres – Apellidos: MARÍA FERNANDA GUILLEN SALAZAR TELMO JAIME BONILLA BONILLA
INFORMACIÓN INSTITUCIONAL
Facultad: INFORMÁTICA Y ELECTRÓNICA
Carrera: ELECTRÓNICA Y AUTOMATIZACIÓN
Título a optar: INGENIERO EN ELECTRÓNICA Y AUTOMATIZACIÓN
f. Analista de Biblioteca responsable: Firmado electrónicamente por: ELIZABETH FERNANDA AREVALO MEDINA

UNIDAD DE PROCESOS TÉCNICOS DBRA DIRECCIÓN DE BIBLIOTECAS Y RECURSOS DEL APRENDIZAJE ESPOCH