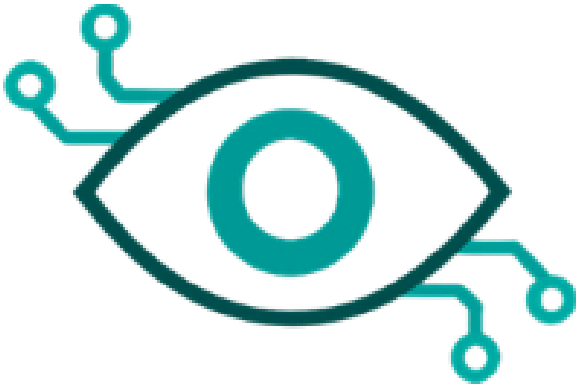
BLINDASSIST

CARPETA TECNICA.

E.E.S.T N°7 TALLER REGIONAL QUILMES
7°2 A.V

Castillo Ramiro.
Quattrocchi Tiago.
Pino Octavio.

Índice



Introducción

1.1. Integrantes

Castillo Ramiro
DNI: 47516171
7°2 Aviónica

Pino Octavio
DNI: 47882634
7°2 Aviónica

Quattrocchi Tiago
DNI: 47510542
7°2 Aviónica

Figura 1.1: Grupo BlindAssist

1.2. Docentes a Cargo

- Argüello Gabriel
- Bianco Carlos
- Carlassara Fabrizio

- Medina Sergio
- Palmieri Diego

1.3. Desarrollo del proyecto

Fecha de Inicio

14/4/2025

Duración del proyecto

25 Semanas

Esfuerzo del proyecto

Maximo de 8 horas de trabajo (200 horas de trabajo en todo el año)

Lenguajes utilizados

- LaTex
- CSS
- HTML
- Python
- Javascript

Programas utilizados

- OverLeaf
- Visual Studio Code
- Github Desktop
- Termius
- Proteus 8
- AutoCad
- UltiMaker
- Benewake Software

Descripción

2.1. Objetivo

El objetivo de este proyecto es mantener la integridad de las personas ciegas o con algún tipo de discapacidad visual en cualquier tipo de entorno urbano y darles un mayor nivel de conciencia en entornos mas cerrados. Para ello, desarrollamos un dispositivo que mediante reconocimiento por inteligencia artificial y la detección de objetos utilizando tecnología láser para avisar al usuario sobre un obstáculo en frente, por encima de la cadera, la presencia de alguna persona en un espacio cerrado y la cantidad de individuos que se encuentran en el mismo espacio, contando con alertas sonoras y vibratorias que no alteren su percepción y condición actual.Este dispositivo funcionara como una ayuda al bastón que utilizan en su día a día sin modificar su efectividad o tratar de reemplazar su función.

2.2. Alcance

- Cómo objetivo final, querríamos que este este proyecto llegue a ser un prototipo funcional y autónomo, con el menor tamaño posible para la comodidad del usuario. Contar con una IA correctamente entrenada y optimizada en rendimiento, que trabaje en conjunto al área cubierta y sensada por los lasers. Teniendo en cuenta las posibilidades, buscamos que las alertas sensoriales sean claras para el usuario así tiene una experiencia placentera en su día a día.

2.3. Temática

- Inclusión social: Nuestro proyecto abarca los parámetros presentados en la temática de inclusión social. Esto debido a que el proyecto mismo busca ayudar con el tránsito de las personas con discapacidad tanto en la vía pública cómo lugares cerrados desconocidos. Además, también puede ayudar con su comunicación y vida social, ya que los alerta de las personas que tienen a su alrededor.

2.4. Problemas a resolver

- Obstáculos repentinos en la vía publica, pueden presentar dificultad a la hora de detectar postes, árboles y otros peligros inesperados que se encuentren por encima de su tren inferior (a partir de la cadera), siendo que la mayoría de las veces el rango que cubre el bastón no es suficiente para detectar y avisar efectivamente a la persona con discapacidad visual.

- El riesgo constante que sufren en la vía publica termina llevando a que dependan mucho de terceros, como puede ser transeúntes que los apoyen o en casos de personas con una edad mas avanzada, un cuidador designado o familiar cercano. Tenemos en cuenta la falta de autonomía que genera esta problemática, la dependencia que terminan teniendo de terceros es una de las problemáticas más importantes a resolver y la idea es darles una herramienta de autonomía e integración.

- Proteger su intimidad pudiendo detectar personas que están en el mismo espacio cerrado sin depender de que la persona avise de su llegada, también permitiéndoles

conocer la ubicación de dichos individuos, dándole un mejor sentido y conciencia situacional.

2.5. Público beneficiado

El proyecto está destinado a personas con impedimentos visuales. En el mundo hay 2200 millones de personas con visibilidad reducida, 43,3 millones son no videntes. Esta es la demográfica que será beneficiada directamente por el proyecto, habilitando el tránsito por la vía pública de manera más segura y sin problemas que puedan llegar a afectar su integridad o la rehabilitación a la que están sometidos. Además, al ser un accesorio portable, compacto y difícil de perder u olvidar, no interrumpe en las acciones que pueda realizar el usuario en su vida cotidiana. Este proyecto también beneficiará indirectamente a instituciones privadas como seguros de salud y ART (Aseguradoras de Riesgo del Trabajo), ya que estos pueden ofrecerle como una ayuda al grupo de personas antes descritas. También aplica para la vida cotidiana o la vida pública, ya que se reduce el riesgo de accidentes y la carga que supone a los familiares a la hora de ayudarlo.

Figura 2.1: Tabla de población ciega por continentes.

2.6. Impacto

- En un contexto social, este prototipo tiene el potencial de transformar la vida de millones de personas. Como mencionamos anteriormente, más de 40 millones de personas en todo el mundo viven con discapacidad visual. Este impacto no solo afectaría a las personas con esta discapacidad, sino también a sus cuidadores y familiares directos. Es decir, podría beneficiar a una gran cantidad de personas y comunidades. Especialmente en regiones como África, Asia y América del Sur, donde existe un alto número de personas con discapacidad visual, pero donde los

recursos y la infraestructura necesarios para facilitar su movilidad y vida cotidiana son limitados.

2.7. Descripción General

- BlindAssist es un proyecto que busca facilitar el tránsito de sus usuarios en la vida cotidiana. A través de un dispositivo colgante que puedan llevar encima mismos o en algún accesorio que mejore su movilidad. Esto lo lograríamos con un conjunto de componentes que trabajan en coordinación. Primero una cámara que detecta elementos específicos, por poner un ejemplo: cestos de basura, señalizaciones, etc. Segundo, unos sensores láser que detectan la proximidad de obstáculos cómo ramas u postes. Cómo tercero, alertas sensoriales a través de audio y vibraciones para alertar al usuario y que este pueda evitar estos inconvenientes.

2.8. Funcionalidades

- El dispositivo presenta dos etapas de detección y dos de alerta al usuario. El dispositivo realizará mediciones de distancia mediante los sensores TFmini LIDAR, estos se encuentran dispuestos de tal forma que puedan cubrir varios ángulos en frente de la persona. Acorde a la detección de los sensores, se le comunica al usuario la posición de un objeto intruso en el rango frontal, dependiendo que sensores detectan un objeto a menor distancia que 120cm se realizara un control de potencia de los motores internos, hay dos motores de vibración, uno a la izquierda y derecha del dispositivo, cada uno acorde a los sensores LIDAR de ambos lados, Cuando se detecta algo en el tercer TFmini cambia la organización, ambos motores emitirán la misma potencia, en caso de que el motor frontal y lateral detecten algo dicho lado tendrá un control titilante de la vibración, en caso de que los tres sensores se activen, ambos motores vibran al máximo de potencia.
  El sistema de detección basado en Inteligencia Artificial (IA), utiliza la cámara para capturar imágenes continuamente. Un modelo de IA avanzado (YOLOv8) actúa como el cerebro, escaneando cada imagen a alta velocidad para identificar y clasificar objetos relevantes para la seguridad, como personas, autos, bicicletas, camiones y objetos de tropiezo (botellas, etc.). Solo se anuncia un objeto si la IA está lo suficientemente segura de su clasificación (más del 60 % de confianza). La comunicación se realiza mediante voz asíncrona. Esta es la clave de la velocidad: el mensaje de voz (“Una persona”, ”3 autos”, ”Moto fuera”) se envía a un proceso en segundo plano, permitiendo que la cámara y la IA sigan escaneando el entorno sin detenerse ni un instante. Para evitar el ruido constante, un filtro de tiempo de espera (cooldown) asegura que solo se anuncien los cambios de estado (algo nuevo apareció, se fue o cambió de cantidad) y no se repita la misma alerta cada pocos segundos.

Hardware

3.1. Raspberry Pi 4

La Raspberry Pi 4 es una computadora de placa única (SBC) compacta y de bajo consumo que sirve como el çerebro"de su proyecto. A diferencia de las computadoras de escritorio tradicionales, está diseñada para tareas especializadas y prototipado. Utiliza un procesador Broadcom BCM2711 de cuatro núcleos y una arquitectura ARM, lo que la hace ideal para aplicaciones de robótica y automatización. Su funcionamiento se basa en la ejecución de un sistema operativo, como Raspberry Pi OS, que permite programar y controlar los componentes conectados a sus pines de GPIO (General Purpose Input/Output).

Figura 3.1: Modulo Raspberry PI 4

3.1.1. PWM

El PWM (Pulse Width Modulation) en la Raspberry Pi 4 es una técnica digital para generar señales moduladas en ancho de pulso que permiten controlar la potencia aplicada a dispositivos electrónicos. En este modelo de placa, el PWM puede implementarse tanto por hardware (pines GPIO dedicados) como por software (emulación mediante librerías como RPi.GPIO o pigpio).

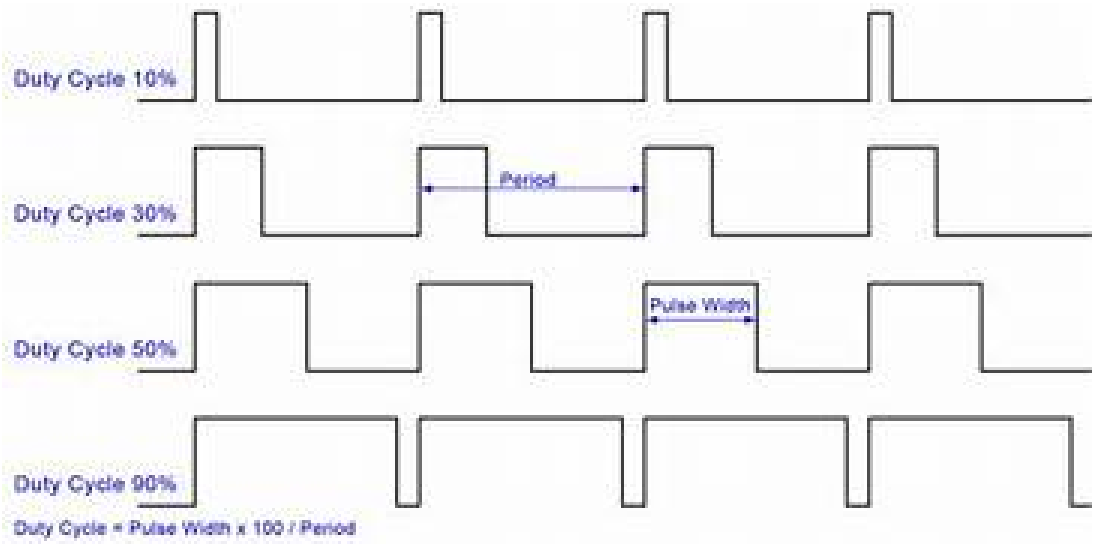


Figura 3.2: Duty cicle de PWM

3.1.1.1. Funcionamiento El PWM consiste en la generación de una onda cuadrada con dos parámetros principales:

- Frecuencia (f): cantidad de ciclos por segundo (Hz).
- Duty cycle (%): proporción del tiempo en que la señal está en estado alto respecto del periodo total.

  La potencia media entregada a la carga es proporcional al duty cycle.

Ejemplo:

- 0 % duty → salida siempre en nivel bajo (0 V).
- 50 % duty → salida mitad del tiempo en nivel alto.
- 100 % duty → salida siempre en nivel alto (3,3 V en GPIO).

3.1.1.2. Pines PWM en Raspberry Pi 4 La Raspberry Pi 4 posee 2 canales PWM de hardware, cada uno con salidas asignables a diferentes GPIO según la configuración. Algunos pines disponibles:

- Canal PWM0: GPIO 12 (pin 32), GPIO 18 (pin 12).
- Canal PWM1: GPIO 13 (pin 33), GPIO 19 (pin 35).

3.1.2. UART

El UART (Universal Asynchronous Receiver-Transmitter) es un protocolo de comunicación serie asíncrono que transmite y recibe datos bit a bit mediante dos líneas:

- TX (Transmit): envía datos.
- RX (Receive): recibe datos.

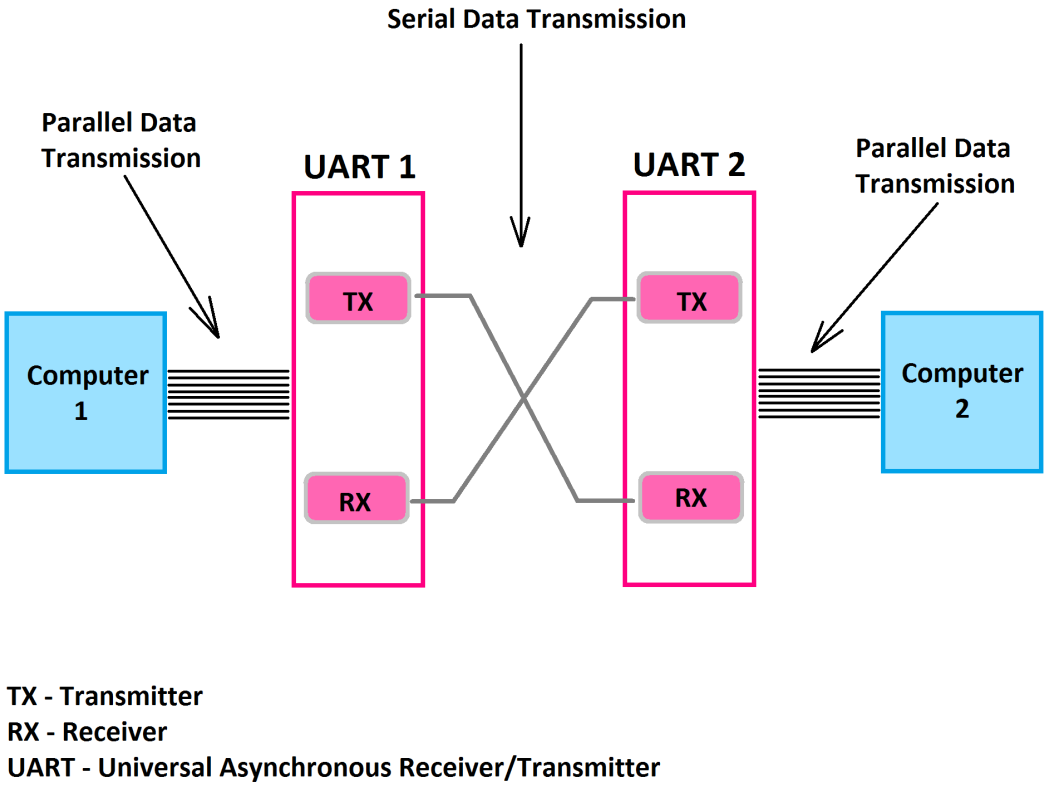


Figura 3.3: Conección UART

En la Raspberry Pi 4, el UART permite la comunicación directa con módulos, microcontroladores, sensores y periféricos externos, usando niveles lógicos de 3,3 V.

3.1.2.1. Funcionamiento La transmisión UART se basa en una trama de datos que incluye:

- Bit de inicio (Start bit).
- Datos (5 a 8 bits).
- Bit de paridad (opcional).
- Bit(es) de parada (Stop bit).

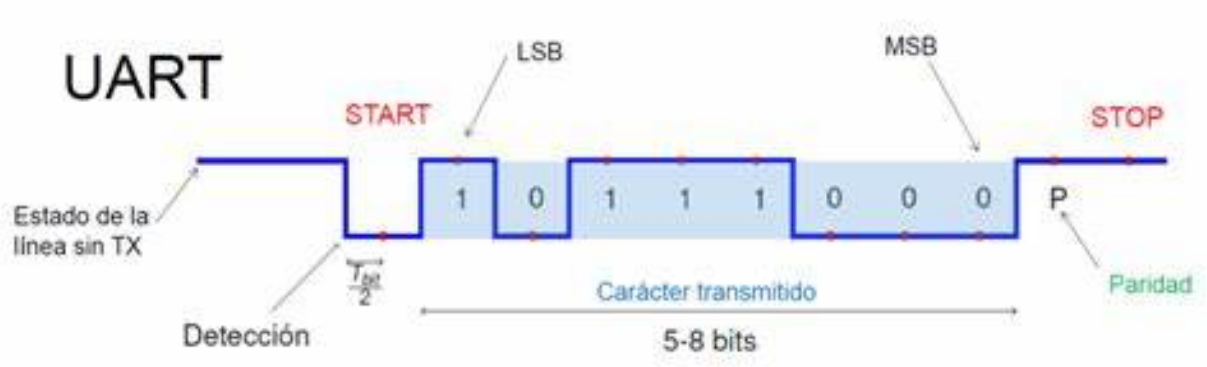


Figura 3.4: Transmisión de Bits en UART

El reloj no se transmite; la sincronización se logra configurando ambas partes con la misma tasa de baudios (baud rate), por ejemplo 9600, 115200 bps.

Es asíncrono, no requiere señal de reloj adicional como en SPI o I²C.

3.1.2.2. UARTs disponibles en Raspberry Pi 4 La placa incluye varios controladores UART, aunque no todos están habilitados de forma predeterminada:

- PL011 (UART principal): más robusto, configurable. Por defecto está mapeado en los pines GPIO14 (TX) y GPIO15 (RX).
- Mini UART: secundario, con funciones más limitadas.

Pines UART por defecto (GPIO – Raspberry Pi 4):

- GPIO14 (TXD0) → Pin físico 8
- GPIO15 (RXD0) → Pin físico 10

3.1.3. Esquemático

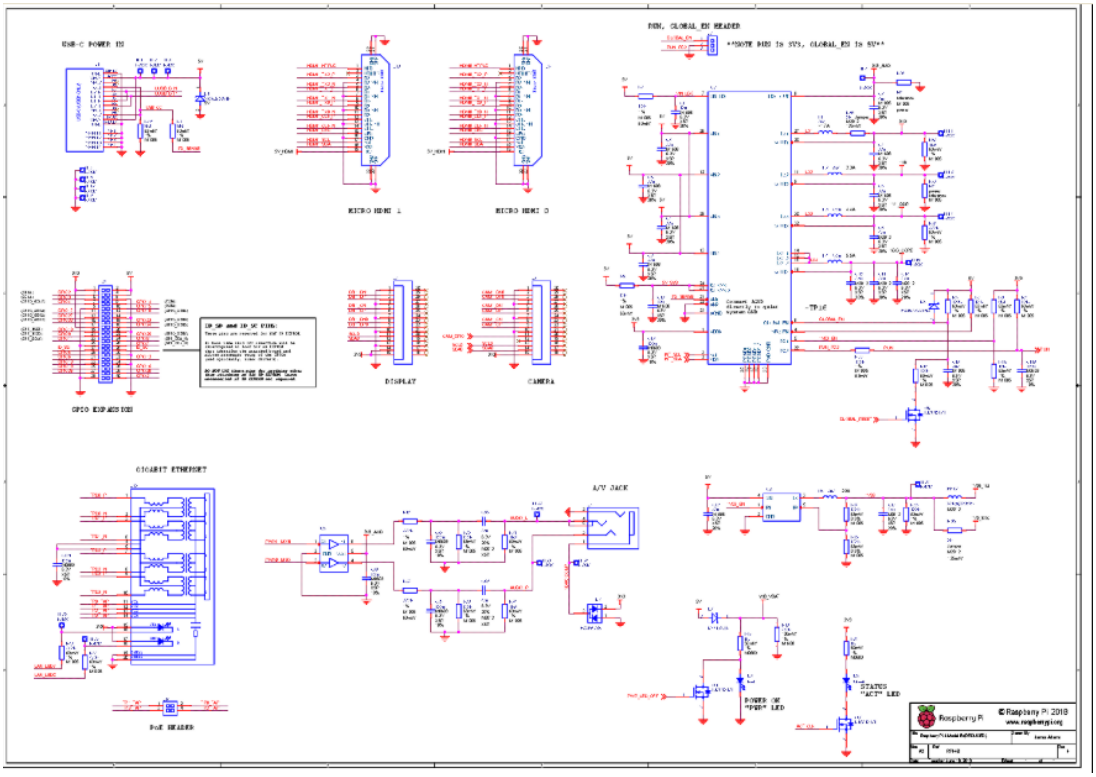


Figura 3.5: Esquematico de la Raspberry PI 4

3.1.4. Pinout

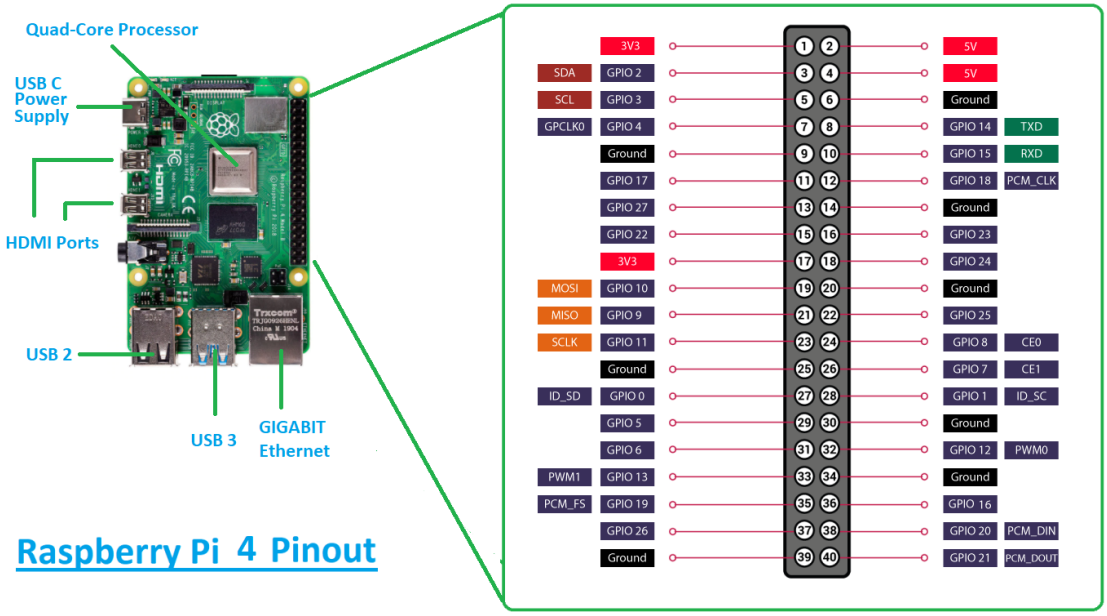


Figura 3.6: Pinout de la Raspberry PI4

3.2. Raspberry PI CAM V1.3

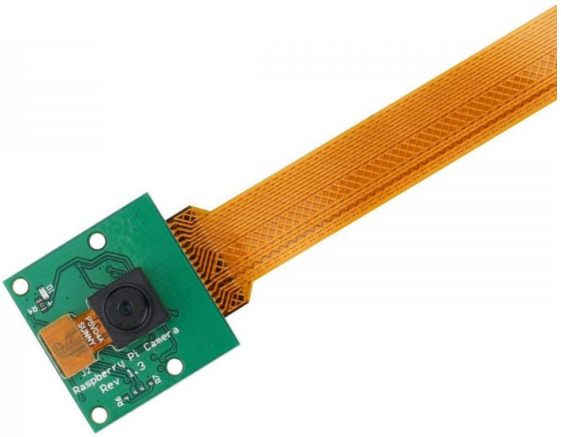


Figura 3.7: Raspberry PI CAM V1.3

La Raspberry Pi Camera Module v1.3 es una cámara oficial diseñada para las placas Raspberry Pi, basada en el sensor OmniVision OV5647 de 5 megapíxeles. Permite capturar

imágenes fijas de hasta 2592×1944 píxeles y grabar video en resoluciones de 1080p a 30 fps, 720p a 60 fps y 480p a 90 fps. Se conecta mediante un cable plano CSI (Camera Serial Interface), lo que asegura una transmisión de datos rápida y optimizada.

Este módulo es compacto y ligero, ideal para visión por computadora, robótica, monitoreo y aplicaciones de IoT. Gracias a su integración directa con la Raspberry Pi y la compatibilidad con librerías como Picamera y OpenCV.

3.2.1. Tablas de parametros

Parámetro	**Descripción**
Tipo de sensor	OmniVision OV5647 Color CMOS QSXGA (5 megapíxeles).
Tamaño del sensor	3,67 x 2,74 mm.
Contenido de píxeles	2592 x 1944 píxeles.
Tamaño de píxel	1,4 x 1,4 µm.
Tamaño de la placa	25 x 24 mm (sin cable flexible).

Cuadro 3.1: Tabla de especificaciones técnicas

Parámetro	**Descripción**
Objetivo	f = 3,6 mm.
Abertura	f/2,9.
Longitud focal	3,29 mm.
Ángulo de visión	54° (horizontal) x 41° (vertical).
Campo de visión total	72,4°.
Campo de visión a 2 m	2,0 x 1,33 m.
Equivalencia	Lente SLR de 35 mm en formato completo.
Enfoque	Fijo, desde 1 m hasta infinito.

Cuadro 3.2: Tabla de parámetros ópticos

3.2.2. Rendimiento Fotográfico

Parámetro	**Descripción**
Disparo mínimo	1/4 s a 5 m.
Resolución de imagen fija	Hasta 2592 x 1944 px (5 MP).

Cuadro 3.3: Tabla de rendimiento fotográfico

Parámetro	**Descripción**
Vídeo Full HD	1080p a 30 fps (códec H.264/AVC).
Vídeo HD	720p a 60 fps.
Vídeo VGA	Hasta 90 fps.

Cuadro 3.4: Tabla de rendimiento de video

3.3. TFmini

El TFmini es un sensor de distancia LiDAR (Light Detection and Ranging). Funciona emitiendo un pulso de láser infrarrojo y midiendo el tiempo que tarda la luz en rebotar en un objeto y regresar al sensor. Esta técnica, conocida como tiempo de vuelo (ToF), le permite calcular la distancia con precisión. Se comunica con la Raspberry Pi a través de interfaces como UART (puerto serie) o I2C, lo que le permite obtener lecturas de distancia de forma rápida y confiable.

Figura 3.8: TFmini S

3.3.1. Descripción General y Principio de Funcionamiento

El TFmini es un mini módulo LiDAR diseñado para realizar funciones de medición de distancia sin contacto y en tiempo real. Se caracteriza por ofrecer una medición de distancia precisa, estable y de alta velocidad.

El sensor opera bajo el principio de TOF (Time of Flight). El módulo transmite periódicamente una onda de modulación de rayo infrarrojo cercano que se refleja al contactar un objeto. El producto calcula el rango relativo al objeto al medir la diferencia de fase de ida y vuelta (tiempo de vuelo) de la señal.

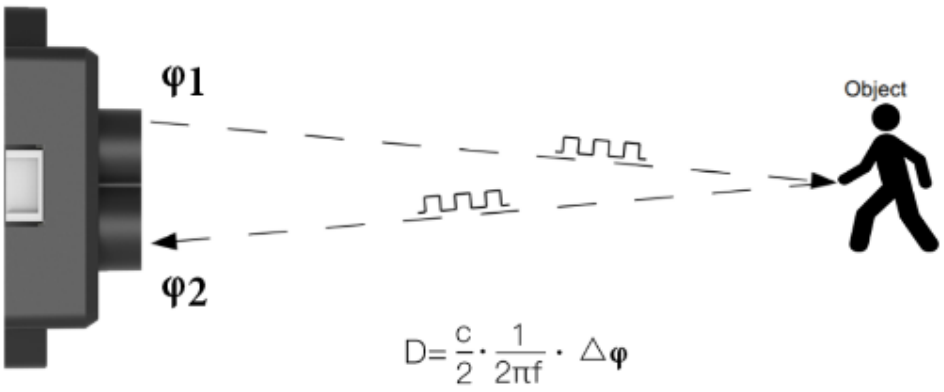


Figura 3.9: Principio de tiempo de vuelo

3.3.2. Tablas de parametros

Característica	Parámetro	Notas
Rango de Operación (Interior)	0.3 m – 12 m	Alcance máximo bajo condiciones de interior y pizarra blanca estándar (reflectividad del 90 %).
Precisión de Medición	±6 cm (@ 0.3 m - 6 m) / ±1 % (@ 6 m - 12 m)	
Frecuencia (Tasa de Salida)	100 Hz	Una salida de datos cada 10 ms.
Resolución de Rango	1 cm	
Unidad de Distancia por Defecto	cm (centímetros)	Se puede modificar a mm (milímetros) mediante instrucción.
Zona Ciega de Detección	0 - 30 cm	Los datos en este rango no se consideran fiables.
Ángulo Medio de Recepción (β)	1.15°	
Ángulo Medio de Transmisión	1.5°	

Cuadro 3.5: Tabla de parámetros y rendimiento

El rango puede verse afectado por la intensidad de la iluminación ambiental y la reflectividad del objeto. Por ejemplo, en condiciones de deslumbramiento extremo (exterior a mediodía de verano, ≈ 100 klux) y detección de un objetivo negro (reflectividad del 10 %), el rango efectivo es de 0.3 m a 3 m.

Característica	Parámetro
Voltaje de Alimentación	5 V
Corriente Promedio	$\leq$ 140 mA
Corriente Pico	800 mA
Potencia Promedio	$\leq$ 0,7 W
Nivel de Comunicación	LVTTL (3.3V)

Cuadro 3.6: Tabla de caracteristicas electricas

El TFmini utiliza el protocolo de comunicación de **puerto serie (UART)**.

Protocolo de Comunicación (UART)	Parámetro por Defecto
Velocidad de Transmisión (Baud rate)	115200
Bit de Datos	8
Bit de Parada	1
Chequeo de Paridad	Ninguno

Cuadro 3.7: Tabla de Interfaz y protocolo de comunicación

Terminal de Conexión Tipo: GH1.25-4P SMT.
Pines: GND (Tierra), +5V (Alimentación), RXD (Recepción), TXD (Transmisión).

Formatos de Salida de Datos

- Formato Estándar (por defecto): Paquete de 9 bytes en formato hexadecimal (HEX). Incluye Distancia (Dist), Intensidad de Señal (Strength), Modo de Distancia (Mode) y Byte de Chequeo (CheckSum).
- Formato de Datos Pixhawk: La salida es una cadena de caracteres (string) con la unidad en metros (m). Se utiliza principalmente para la conexión con Pixhawk.

3.3.3. Estructura y Dimensiones

- Carcasa: Fabricada en ABS + PC.
- Componentes Externos Clave: Lente de transmisión y Lente de recepción.
- Placa de Circuito: La placa de circuito está expuesta en la parte trasera del producto.
- Agujero de Montaje: Agujero pasante de 2.35 mm.
- Dimensiones : 42 mm (largo) x 15 mm (ancho).

Figura 3.10: Proporciones del modulo

3.4. Conversor TTL a USB

El CP2102 es un controlador de puente USB a UART altamente integrado que permite la conversión directa entre interfaces USB y serie. Este dispositivo es ideal para actualizar sistemas que originalmente utilizaban comunicación RS-232, TTL o RS-485, eliminando la necesidad de puertos seriales tradicionales en PCs modernas. Gracias a su tamaño compacto y a la mínima cantidad de componentes externos necesarios, el CP2102 ofrece una solución eficiente, confiable y económica para la comunicación entre ordenadores y dispositivos embebidos.

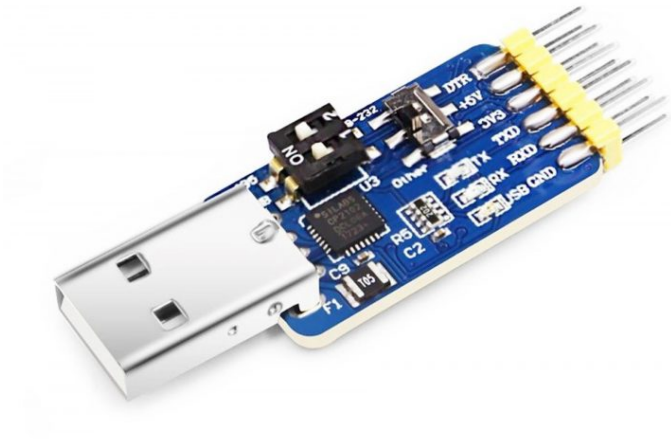


Figura 3.11: Conversor Serie TTL a USB

3.4.1. Funcionamiento

El CP2102 actúa como un puente entre la interfaz USB 2.0 y los protocolos UART, RS-232 o RS-485 del dispositivo conectado. El chip convierte los datos USB en señales

serie y viceversa, facilitando la transmisión y recepción asíncrona de datos. El módulo no requiere componentes externos USB y es reconocido automáticamente por la mayoría de los sistemas operativos mediante drivers integrados.

Parámetro	**Valor / Descripción**
Chip principal	CP2102 – Puente USB - UART
Interfaz USB	USB 2.0 (Full Speed, 12 Mbps)
Protocolos soportados	TTL, RS-232, RS-485
Modos de conversión	USB-TTL, USB-RS232, USB-RS485, TTL-RS232, TTL-RS485, RS232-RS485
Tamaño	32 mm × 18 mm
Voltajes soportados	3.3 V / 5 V
Salidas de potencia	5 V – 500 mA / 3.3 V – 100 mA
Velocidad máxima	Hasta 2 Mbps
Compatibilidad MCU	8051, STC, AVR, ARM, STM32, DSP, FPGA, MSP430, etc.
Señales adicionales	DTR, RTS (descarga BSL para MSP430)
Protección	Fusible autorrearmable ante sobrecorriente (¿500 mA)

Cuadro 3.8: Tabla de características técnicas

3.4.2. Funciones Destacadas

- Módulo 6 en 1: Conversión bidireccional entre múltiples interfaces serie.
- Compatibilidad con 3.3 V y 5 V: Ideal para microcontroladores modernos de bajo voltaje.
- Salidas de alimentación integradas: Posibilidad de alimentar el sistema objetivo durante la depuración.
- Protección contra sobrecorriente: Fusible de recuperación automática integrado.
- Indicadores LED: Visualización del flujo de datos en transmisión y recepción.
- Control automático RS-485: No requiere líneas adicionales de control.
- Transferencia libre entre protocolos: Conmutación física confiable sin interferencias.

3.5. Fuente Step-down

Un buck converter (convertidor reductor) es un convertidor DC-DC que reduce una tensión de entrada (Vin) a un valor menor de salida (Vout), manteniendo una alta eficiencia gracias al uso de conmutación en lugar de disipación resistiva.

En este caso, el regulador está basado en el XL4015, un chip controlador step-down con capacidad de hasta 5 A y frecuencia de trabajo en torno a 180 kHz.

Figura 3.12: Fuente buck step-down 5V

3.5.1. Funcionamiento

El XL4015 regula la salida usando PWM de alta frecuencia y los siguientes componentes externos:

- MOSFET interno (dentro del XL4015): conmuta la corriente desde la entrada hacia la bobina.
- Inductor (L1): almacena energía en forma de campo magnético y la entrega suavizada a la carga.
- Diodo Schottky (D1, SS54): permite el flujo de corriente cuando el MOSFET está apagado, garantizando continuidad de corriente.
- Capacitores (C1, C2, C3, C4, C5): filtran el rizado de entrada y salida.
- Resistencias R1 y R2 (red de realimentación): forman un divisor que regula la tensión de salida comparándola con la referencia interna del chip.
- R3 y C6: compensación y estabilidad del lazo de control.
- Potenciómetro: ajusta la tensión de salida.

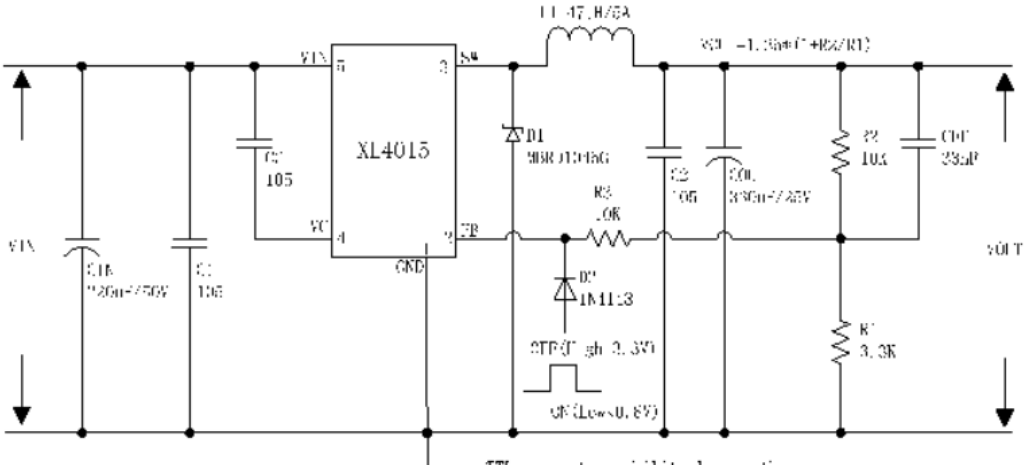


Figura 3.13: Esquemático de Fuente buck step-down

3.5.2. Etapas

1. Fase de Conducción (MOSFET ON)

   El MOSFET interno del XL4015 se enciende. La corriente fluye desde Vin → L1 → C4/C5 → Carga. La bobina almacena energía en su campo magnético, aumentando su corriente.

2. Fase de Retención (MOSFET OFF)

   El MOSFET se apaga. La corriente en el inductor no puede caer abruptamente, por lo que el diodo D1 conduce. La energía almacenada en L1 se transfiere hacia la carga y capacitores de salida.

3. Regulación

   El divisor resistivo (R1, R2) mide la tensión de salida. Esta señal se compara con una referencia interna (1,25 V típica). El controlador ajusta el duty cycle del PWM del MOSFET interno para mantener la salida estable.

3.5.3. Parámetros del Esquema

- Chip: XL4015 (regulador step-down DC-DC, hasta 5A).
- Diodo: SS54 (Schottky, rápido y baja caída de tensión).
- Inductor: 47 µH, soporta varios amperios.
- Resistencias: R1 (0,05 Ω detecta corriente, R2 ajusta Vout.
- Capacitores:
  - Entrada: 220 µF (electrolítico) + 0,1 µF (cerámico) → filtran ruido y picos de corriente.
  - Salida: 220 µF + 0,1 µF → reducen rizado en Vout.

3.6. Baterías

Las baterías Li-Ion cilíndricas tipo 18650 de 3.7 V y 7800 mAh son acumuladores recargables de alta capacidad, diseñados para aplicaciones que requieren energía confiable, autonomía prolongada y recarga eficiente. Su formato estandarizado las hace ampliamente compatibles con dispositivos electrónicos de consumo y proyectos de ingeniería. Figura 3.14: Baterías de litio

Figura 3.14: Baterías recargables

3.6.1. Características Técnicas

- Química: Ion de Litio (Li-ion).
- Formato: Cilíndrico 18650 (18 mm diámetro × 65 mm longitud).
- Voltaje nominal: 3,7 V.
- Voltaje máximo de carga: 4,2 V.
- Voltaje de corte (descarga): 2,5–3,0 V.
- Capacidad nominal: 7800 mAh (7,8 Ah).
- Energía aproximada: 28,9 Wh por celda.
- Conectividad: Terminales planos, aptos para soldadura o porta pilas.

3.7. Motores de vibración

Los motores de vibración que vamos a utilizar son motores de corriente continua (DC) conocidos como Eccentric Rotating Mass (ERM). Su principio de funcionamiento es muy sencillo pero efectivo: El motor tiene un eje que gira cuando se le aplica energía eléctrica. Lo que crea la vibración es un pequeño peso o masa que está unido al eje, pero no en su centro. Este desplazamiento del centro de masa hace que la rotación sea desequilibrada. A medida que el motor gira a alta velocidad, la masa descentrada genera una fuerza centrífuga que cambia de dirección constantemente. Este movimiento rápido y desequilibrado es lo que produce la vibración que se siente. En muchos dispositivos como es el caso de blindassist que requieren retroalimentación háptica, se usan comúnmente dos motores ERM de diferentes tamaños. Esto permite generar distintos niveles de vibración, desde un temblor fuerte y notorio hasta una vibración suave y sutil, mejorando la experiencia del usuario.

Figura 3.15: Motores de vibración

3.8. Circuito

3.8.1. Lista de Componentes

Componente	Cantidad	Descripción / Función
Raspberry Pi 4 (J5)	1	Microprocesador central, adquisición de datos y control
Sensor TFmini (J1, J2, J3)	3	LIDAR para medir distancias
Motores de vibración (J6, J7)	2	Generación de señal haptica de alerta
Transistor BC337 (Q1, Q2)	2	Etapa de potencia para buzzers
Resistencias 510 Ω (R1, R2)	2	Limitación de corriente en base de transistores
Interruptores (J4, J8)	2	Habilitación/deshabilitación manual
Alimentación de baterias (J9, J10)	2	Alimentacion de la Raspberry Pi 4 y de los motores de vibración)

Cuadro 3.9: Tabla de la lista de componentes del circuito

3.8.2. Descripción del circuito

1. Los sensores TFmini (J1, J2, J3) envían datos de distancia vía UART a la Raspberry Pi 4 (J5).
2. La Raspberry Pi recibe y procesa la información. Si la distancia medida es menor a un umbral, activa una salida GPIO.
3. El GPIO activa Q1 o Q2 mediante las resistencias R1 y R2.
4. Los transistores permiten el paso de corriente hacia los motores (J6, J7), que generan la alerta haptica de vibración.
5. Los interruptores J4 y J8 permiten habilitar/deshabilitar la deteccion de la IA y seleccionar modos de funcionamiento.

3.8.3. Esquematico

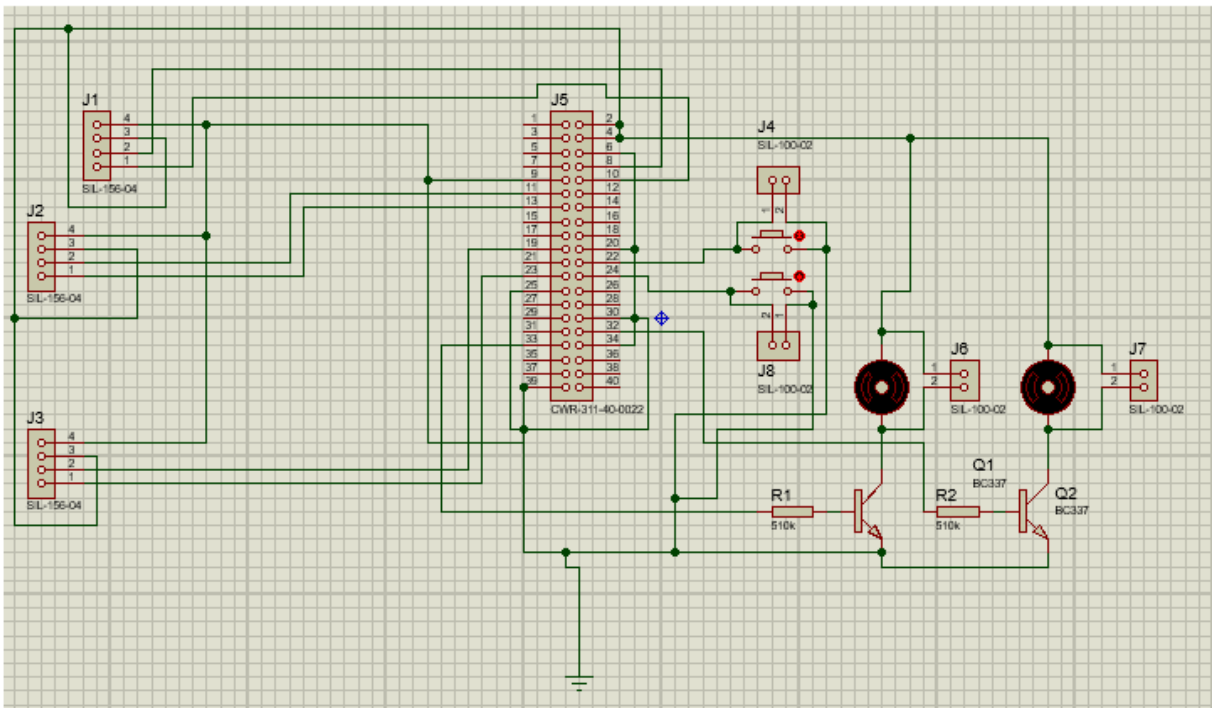


Figura 3.16: Esquematico del circuito final

3.8.4. PCB

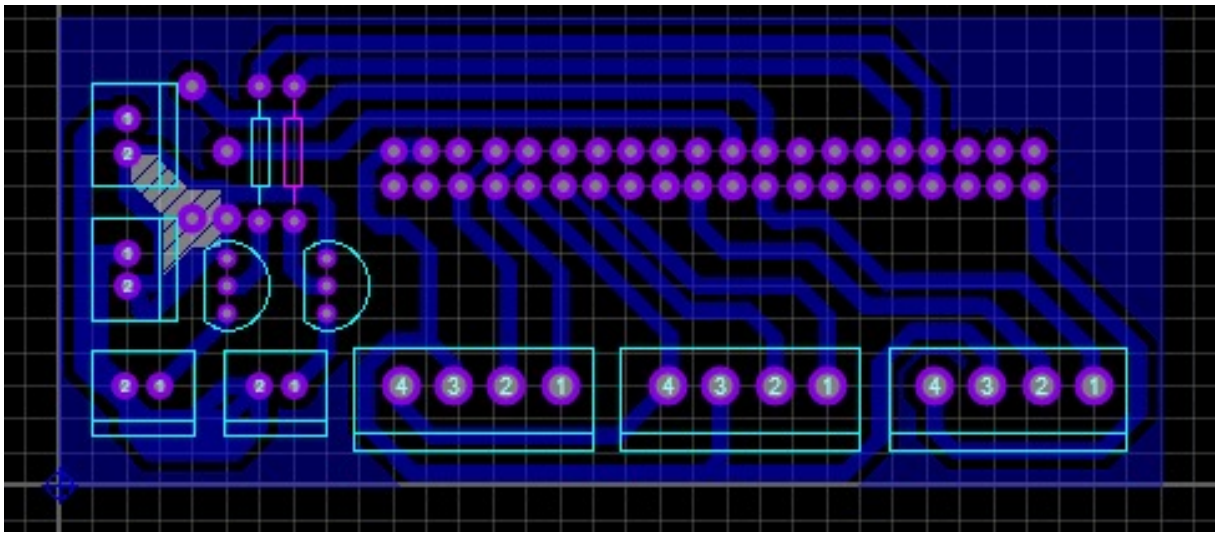


Figura 3.17: diseño de impresion del circuito

Firmware

4.1. Detección por IA

El sistema de visión artificial implementado en este proyecto se basa en el modelo de aprendizaje profundo YOLOv8n.pt para la identificación de objetos en tiempo real, utilizando la cámara de la Raspberry Pi como fuente de datos. La arquitectura del software se estructura en un sistema de Programación Orientada a Objetos (POO). Cada componente lógico (cámara, IA, voz, traductor) está encapsulado en una clase dedicada, lo que facilita el desarrollo, las pruebas y el mantenimiento del sistema de manera modular.

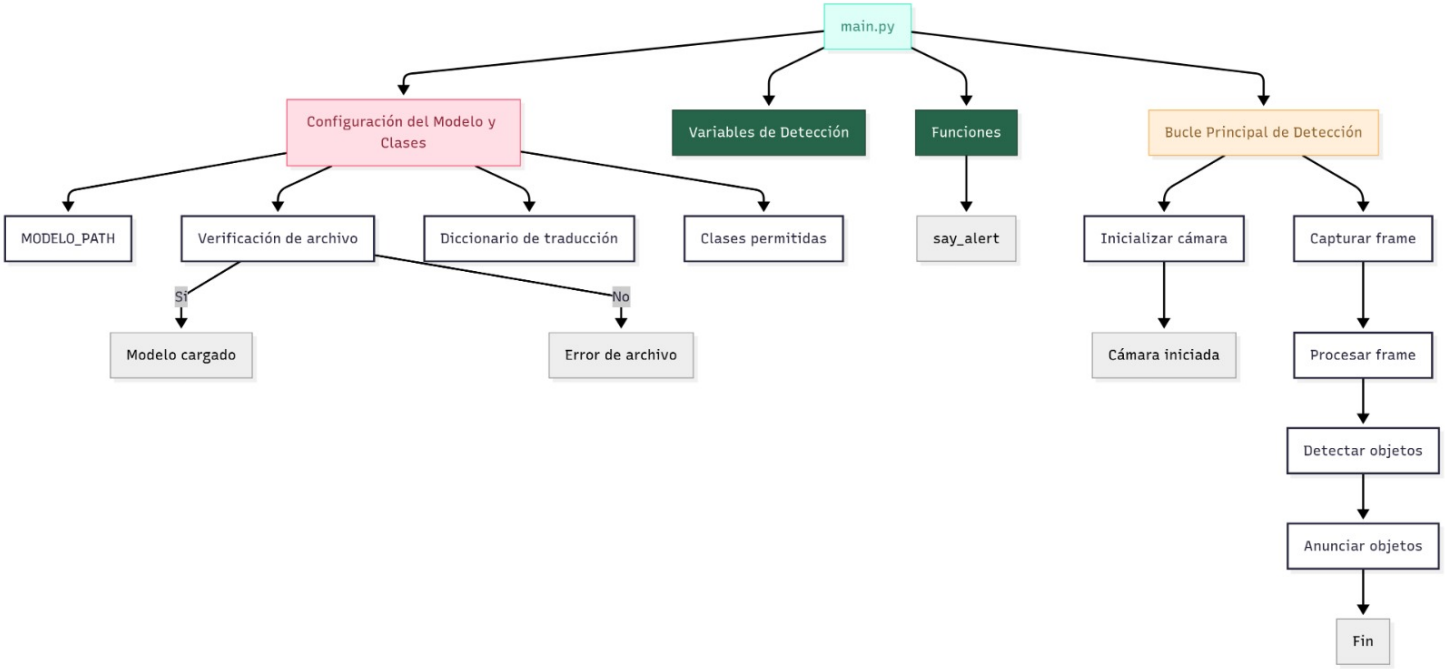


Figura 4.1: Diagrama de flujo de la deteccion por IA

4.1.1. logica del bucle continuo

- Captura frames de video.
- Analiza cada frame en busca de objetos.
- Compara las detecciones con las del frame anterior.
- Emite una alerta de voz solo cuando se detecta un cambio en el entorno (aparición o desaparición de un objeto).

4.1.2. Arquitectura del Código

El programa se organiza en un conjunto de clases principales:

- **Clase Camara:** Gestiona el hardware de la cámara. Utiliza la librería picamera2 para inicializar y capturar un stream de video en vivo. A diferencia de soluciones de un solo fotograma, esta aproximación optimiza el rendimiento y la tasa de fotogramas por segundo (FPS) al mantener el stream de video abierto.
- **Clase IA:** Actúa como el motor de inteligencia artificial. Carga el modelo yolov8n.pt y es responsable de analizar los fotogramas capturados. Identifica objetos en una lista predefinida de clases (clases_permitidas) y filtra las detecciones por un umbral de confianza (CONFIDENCE_THRESHOLD), asegurando que solo los objetos relevantes y con alta certeza sean considerados.
- **Clase Traductor:** Encapsula el diccionario de traducciones. Su única responsabilidad es convertir los nombres de las clases de YOLO (en inglés, como 'person' o 'car') a sus equivalentes en español, permitiendo que el sistema de voz dé las alertas en el idioma del usuario.

- **Clase Voz:** Maneja la síntesis de voz. Para garantizar la fiabilidad del audio, utiliza un método robusto basado en la librería subprocess para ejecutar el comando del sistema espeak y vocalizar las alertas. Esta solución es inmune a los problemas de configuración de audio que se presentan a menudo con otras librerías.
- **Clase Main:** Es el orquestador principal del sistema. Inicializa todas las demás clases y gestiona el bucle de detección en tiempo real. Su lógica más avanzada reside en el método privado __anunciar_detecciones, que compara las detecciones actuales con las del último ciclo, emitiendo una alerta de voz únicamente cuando se detecta un cambio. Esto evita la repetición constante de las mismas alertas.

El sistema se ejecuta continuamente en un bucle while hasta que el usuario lo interrumpe con Ctrl + C. En ese momento, un bloque finally asegura que el hardware de la cámara se detenga de manera segura para evitar fallos del sistema.

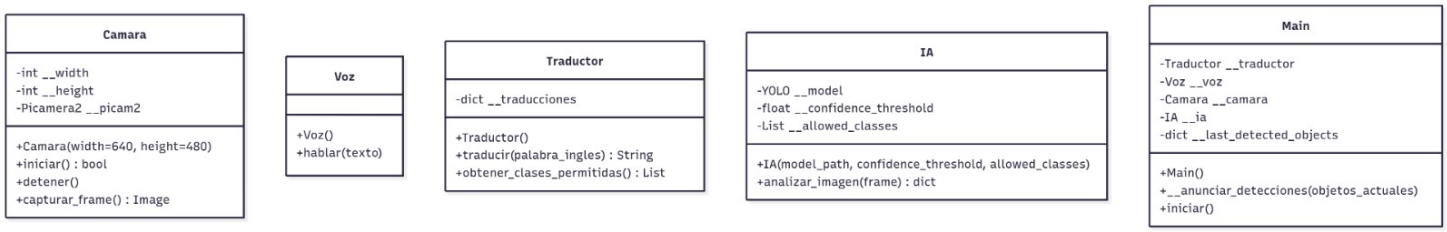


Figura 4.2: Diagrama de flujo de clases

4.1.3. Componentes de la Implementación

4.1.3.1. Librerías

- ultralytics: Se utiliza para cargar y ejecutar el modelo de detección de objetos YOLOv8.
- picamera2: La interfaz oficial para las cámaras de la Raspberry Pi, proporcionando acceso a un flujo de video de alto rendimiento.
- cv2 (OpenCV): Se usa para el preprocesamiento de los fotogramas, como la conversión de color necesaria antes de pasarlos al modelo de IA.
- subprocess: La herramienta estándar de Python para ejecutar comandos del sistema, en este caso, el motor de síntesis de voz espeak.
- os y sys: Permiten la gestión de archivos y el control del sistema, por ejemplo, para verificar la existencia del modelo YOLO antes de su uso.

4.1.3.2. Manejo de Flujo de Datos El flujo de información y la lógica de control se manejan principalmente en el método iniciar() de la clase Main:

- **Captura:** self.__camara.capturar_frame() obtiene una imagen del stream de la cámara.
- **Análisis:** El frame se pasa a self.__ia.analizar_imagen(frame), que devuelve un diccionario con los objetos detectados en el fotograma actual.

- **Control de Anuncios:** El diccionario se envía al método __anunciar_detecciones().
- **Reproducción de Audio:** Si hay cambios, la clase Voz es invocada a través de self.__voz.hablar(final_message).

La arquitectura del sistema es muy escalable. Al estar modularizado, se podrían añadir fácilmente nuevas clases sin modificar la lógica central del proyecto.

4.2. Detección por láser

4.2.1. Libreria TFmini

Esta sección describe el funcionamiento del código implementado para la lectura simultánea de tres sensores LiDAR TFmini conectados a diferentes puertos UART de una Raspberry Pi 4.

El sistema permite:

- Recepción continua de datos desde los sensores.
- Procesamiento y almacenamiento de las distancias medidas.
- Ejecución en paralelo mediante hilos (multithreading).
- Disponibilidad centralizada de los valores para otras funciones del sistema.

Descripción General del Sistema: El programa se estructura en cuatro bloques principales:

- Configuración global: Definición de librerías, estructuras de datos y variables compartidas.
- Módulo de adquisición de datos: Funciones de lectura desde los sensores a través de los puertos UART.
- Gestión de hilos: Creación de procesos concurrentes para la lectura simultánea.
- Bucle principal: Supervisión e impresión periódica de los valores capturados.

La arquitectura se basa en la comunicación serial estándar (UART) a una velocidad de 115200 baudios, con tramas de 9 bytes emitidas por cada TFmini.

4.2.1.1. Librerías:

- **serial (pyserial):** Permite abrir, configurar y gestionar la comunicación UART.
- **time:** Se utiliza para pausas entre lecturas y evitar sobrecarga del procesador.
- **threading:** Facilita la ejecución concurrente de múltiples lecturas de sensores.

4.2.1.2. Estructuras Globales:

```
distancias = {
"uart1": None,
"uart2": None,
"uart3": None,
}
```

Diccionario que almacena la última distancia válida obtenida de cada sensor. Inicialmente los valores son None y se actualizan en tiempo real. Facilita la consulta de datos desde cualquier parte del programa.

4.2.1.3. Función Genérica de Adquisición de Datos:

```
def getTFminiData(port, key_name):
ser = serial.Serial(port, 115200, timeout=1)
```

Flujo de la función: Apertura del puerto serie con parámetros: Velocidad de 115200 baudios, timeout de 1 segundo. Bucle infinito de lectura: Verifica si hay al menos 9 bytes disponibles (ser.in_waiting ¿8). Lee un paquete completo (ser.read(9)). Validación de cabecera: Primeros dos bytes deben ser 0x59 0x59. Si no coincide, se descarta la trama. Reconstrucción de la distancia: Byte [2] = parte baja (Low). Byte [3] = parte alta (High). Cálculo: distance = low + (high « 8).

»»

4.2.1.4. Almacenamiento de datos La distancia calculada se guarda en el diccionario global bajo la clave correspondiente (uart1, uart2, uart3). Retardo de control time.sleep(0.01) para liberar recursos del procesador.

4.2.1.5. Funciones Específicas por Puerto UART

```
def getTFminiData_uart1():
getTFminiData("/dev/serial0", "uart1")
```

Función getTFminiData_uart1: Llama al puerto /dev/serial0. Función getTFminiData_uart2: Llama al puerto /dev/ttyUSB1. Función getTFminiData_uart3: Llama al puerto /dev/ttyUSB2. Estas funciones permiten escalar el sistema sin modificar la lógica principal.

4.2.1.6. Gestión de Hilos y Ejecución Principal

```
if __name__ == '__main__':
...
```

Creación de hilos: Cada hilo ejecuta una de las funciones específicas (uart1, uart2, uart3). Se utilizan hilos daemon, lo que asegura que se cierren cuando termine el programa. Ejecución concurrente: Los tres hilos inician en paralelo con .start(). Se garantiza la lectura simultánea de los sensores. Bucle principal de supervisión: Cada segundo imprime las distancias almacenadas en distancias y permite monitorear en consola el funcionamiento del sistema. Interrupción del usuario: Al presionar Ctrl + C, se captura la excepción KeyboardInterrupt. Se muestra un mensaje de salida segura.

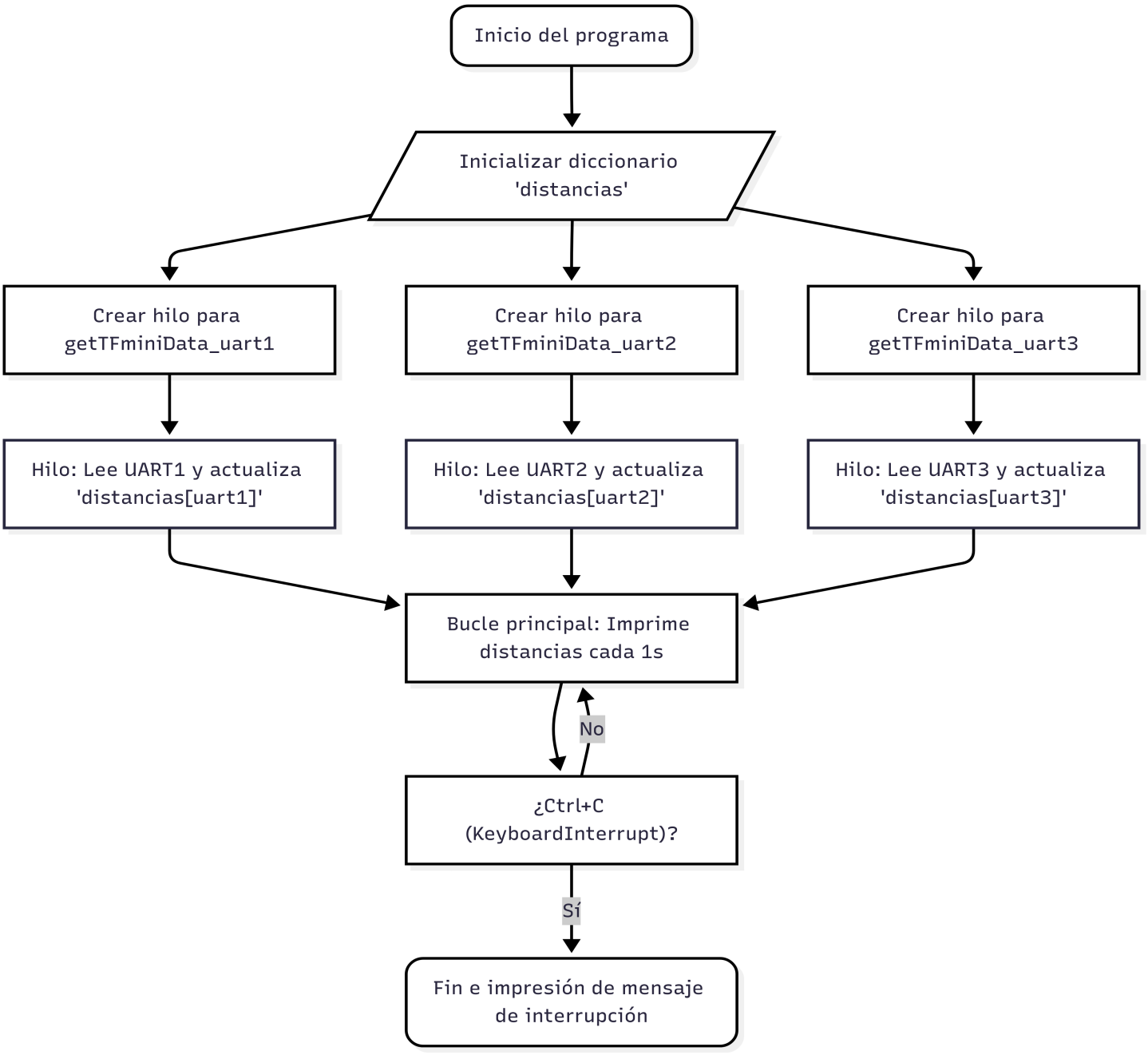


Figura 4.3: Diagrama de flujo de la lectura de distancias

4.2.2. Control de Motores

El código implementa un sistema de control de dos motores mediante modulación por ancho de pulso (PWM) utilizando una Raspberry Pi 4. El control se basa en las distancias medidas por tres sensores LiDAR TFmini conectados a los puertos UART de la Raspberry Pi. El objetivo del programa es variar la velocidad de los motores en función de la proximidad de obstáculos, generando alertas hapticas (mediante vibraciones) cuando se detectan objetos a menos de 120 cm.

4.2.2.1. Librerías y Dependencias

- `time`: Control de tiempos y retardos.
- `threading`: Permite ejecutar lecturas de los tres sensores en paralelo mediante hilos.
- `RPi.GPIO`: Control de los pines GPIO de la Raspberry Pi.
- `TFtest3`: Módulo externo que contiene las funciones de lectura `getTFminiData_uart1/2/3` y el diccionario `distancias`.

4.2.2.2. Configuración del Sistema

- **Modo GPIO:**

```
GPIO.setmode(GPIO.BCM)
```

- **Asignación de Pines PWM:**

```
MOTOR_PIN1 = 18
MOTOR_PIN2 = 13
```

Estos pines son compatibles con PWM por hardware en la Raspberry Pi 4.

- **Inicialización de PWM:**

```
pwm1 = GPIO.PWM(MOTOR_PIN1, 100)
pwm2 = GPIO.PWM(MOTOR_PIN2, 100)
pwm1.start(0)
pwm2.start(0)
```

Se inicializan ambas salidas con una frecuencia de **100 Hz** y un duty cicle del **0 %**.

4.2.2.3. Función Auxiliar

`calcular_duty(d)`: Esta función traduce la distancia medida (en cm) al porcentaje de ciclo útil del PWM. Si el objeto está a menos de **120 cm**, el duty cycle aumenta proporcionalmente al acercamiento.

```
if d < 120:
    return min(100, 120 - d)
```

Ejemplo de funcionamiento:

- Si $d = 100$ cm $\rightarrow$ $duty = 20\,\%$
- Si $d = 50$ cm $\rightarrow$ $duty = 70\,\%$

4.2.2.4. Lógica de Control Principal

Función central que coordina la lectura de los tres sensores y actualiza los dos motores según las condiciones detectadas. **Ciclo de operación:**

1. Reinicia el PWM en 0 % al comienzo de cada ciclo.
2. Lee las distancias `d1`(Sensor derecho), `d2`(Sensor izquierdo), `d3`(Sensor frontal) desde el diccionario compartido `distancias`.
3. Calcula los valores de duty correspondientes.
4. Evalúa las condiciones de proximidad menores a 120 cm.

Casos contemplados:

Caso	Condición	Acción
1	d1 $<$ 120	Activa PWM0 proporcionalmente.
2	d2 $<$ 120	Activa PWM1 proporcionalmente.
3	d3 $<$ 120	Ambos motores actúan con el mismo PWM.
4	d2 y d3 $<$ 120	Motor 2 titila alternadamente.
5	d1 y d3 $<$ 120	Motor 1 titila alternadamente.
6	d1, d2 y d3 $<$ 120	Ambos motores al 100 % (alerta crítica).

Cuadro 4.1: Casos de control de motores según distancias detectadas.

4.2.2.5. Ejecución Principal

El bloque `if __name__ == "__main__":` inicia la ejecución del programa. Se crean tres hilos dedicados a la lectura continua de los sensores TFmini:

```
t1 = threading.Thread(target=getTFminiData_uart1, daemon=True)
```

Un cuarto hilo ejecuta la función `control_sensores()` para el control de los motores. El bucle principal mantiene el programa activo indefinidamente mientras los hilos trabajan en segundo plano.

4.2.2.6. Finalización Segura

En caso de interrupción manual (Ctrl + C), el programa detiene los PWM y limpia la configuración de los GPIO:

```
pwm1.stop()
pwm2.stop()
GPIO.cleanup()
```

Esto evita errores en ejecuciones posteriores.

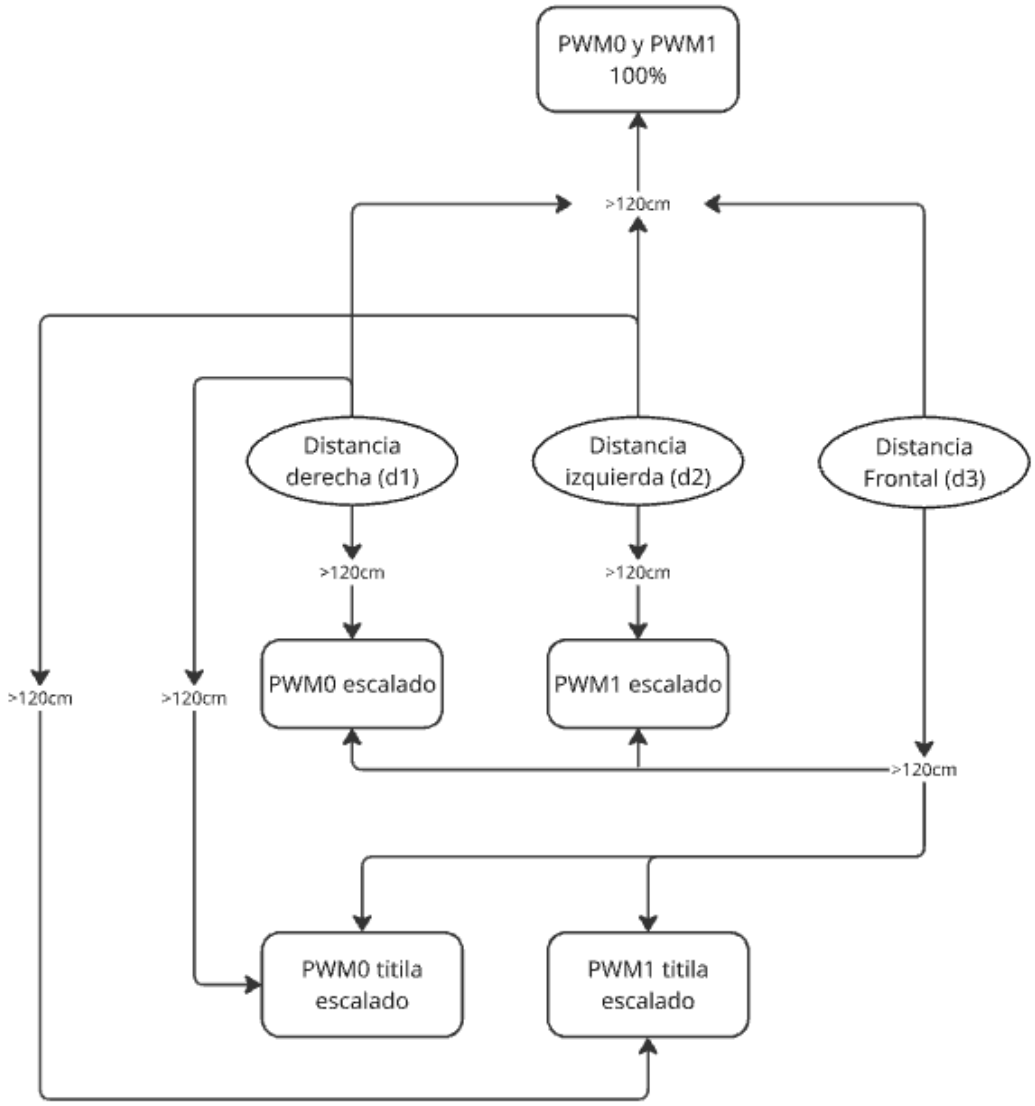


Figura 4.4: Diagrama de flujo e la logica PWM

4.2.3. Código Final del Control Principal (control_principal.py)

La función principal de este script es coordinar todos los módulos de hardware y software: cámara, sensores LiDAR, motores de vibración y el sistema de voz. Gracias a esta integración, el dispositivo puede ofrecer detección de objetos y advertencia de obstáculos de manera simultánea y en tiempo real.

4.2.3.1. Importación de Librerías y Configuración

Al inicio, el código importa distintas librerías necesarias para manejar el hardware y los módulos de inteligencia artificial:

```
import RPi.GPIO as GPIO
import threading
import time
```

```
import cv2
from ultralytics import YOLO
```

En términos generales, estas librerías permiten conectar la Raspberry Pi con el entorno físico (botones, motores, cámara), al mismo tiempo que se integra el modelo de IA encargado de reconocer objetos.

4.2.3.2. Gestión de Pines y Estados Globales

El sistema trabaja con botones y motores conectados a pines GPIO de la Raspberry Pi. Estos pines permiten encender o apagar funciones al pulsar un botón y regular la intensidad de vibración de los motores mediante señales PWM.

```
PIN_DETECCION = 25   # Botón para activar la cámara
PIN_LIDAR = 8        # Botón para activar el LiDAR
MOTOR_PIN1 = 18
MOTOR_PIN2 = 13
```

De este modo, el usuario puede elegir si activa la detección de objetos o el aviso de obstáculos, cada uno ejecutado en hilos separados.

4.2.3.3. Lógica Multihilo (Threads)

El programa aprovecha la librería threading para ejecutar varias tareas al mismo tiempo.

- Al presionar el botón de detección, se inicia un hilo que activa la cámara y el modelo de IA. Este módulo captura imágenes, analiza lo que ve y notifica mediante voz qué objetos han aparecido o desaparecido en el entorno.
- Al presionar el botón de LiDAR, se inician varios hilos que leen la distancia desde los sensores y controlan los motores de vibración. Cuanto más cerca esté un obstáculo, más intensa es la vibración.

De esta forma, el usuario recibe información visual traducida tanto en mensajes de voz como en señales hápticas.

4.2.3.4. Manejo de Eventos con GPIO

El sistema está preparado para reaccionar automáticamente cuando se presiona un botón. Esto se logra configurando los pines de entrada de la Raspberry Pi con interrupciones:

```
GPIO.add_event_detect(PIN_DETECCION, GPIO.FALLING,
                      callback=boton_callback, bouncetime=300)
```

La función `boton_callback()` determina qué botón fue presionado y activa o desactiva el hilo correspondiente. Si un módulo ya estaba en ejecución, el mismo botón permite detenerlo.

4.2.3.5. Función Principal (main) El núcleo del programa es la función `main()`, que inicia el sistema, espera unos segundos para asegurar que todo el hardware esté listo, y finalmente queda a la espera de que el usuario presione algún botón:

```
def main():
    print("[INFO] Esperando 10 segundos...")
    time.sleep(10)
    iniciar_gpio()
    voz = Voz()
    voz.hablar("Sistema listo para operar.")
```

Esto asegura un arranque limpio y confiable.

4.2.3.6. Procedimiento de Despliegue y Arranque Automático

Un aspecto fundamental del proyecto fue garantizar que el sistema se ejecute de forma automática al encender la Raspberry Pi. Para ello, se descartó el uso de `crontab`, que presentaba fallos al iniciar el entorno virtual de Python, y se optó por `systemd`, la herramienta moderna de Linux para gestionar servicios.

4.2.3.7. Archivo de Servicio: blindassist.service

El archivo `blindassist.service`, ubicado en `/etc/systemd/system/`, define cómo y cuándo debe ejecutarse el programa principal. Su contenido clave es:

```
[Unit]
Description=Servicio de Asistencia Visual - Arranque Automatico
After=network.target multi-user.target

[Service]
Type=idle
User=blindassist
ExecStart=/bin/bash -c "sleep 10 &&
/home/blindassist/repo/firmware/.venv/bin/python
/home/blindassist/repo/firmware/control_principal.py"
Restart=always
RestartSec=5

[Install]
WantedBy=multi-user.target
```

Este archivo cumple con tres funciones principales:

- **Arranque Diferido:** El comando `sleep 10` garantiza que el sistema operativo haya terminado de cargar antes de iniciar el script.
- **Ejecución en Segundo Plano:** El programa se ejecuta como un servicio del sistema, independiente de sesiones de usuario.
- **Robustez:** En caso de fallo, el servicio se reinicia automáticamente gracias a las directivas `Restart=always` y `RestartSec=5`.

Diseño

Prototipo 3D

Materiales Utilizados / Programas

- Filamento de impresión 3D PLA.
- Tornillos 1/8
- AutoCAD

5.1. Diseño General

El diseño corresponde a un prototipo de carcasa destinado a integrar de forma eficiente los sensores y componentes del proyecto. Su geometría incluye un frente inclinado en forma triangular, que permite optimizar los ángulos de medición de los LIDAR y ampliar su campo de detección en entornos urbanos. También incorpora un orificio frontal para la cámara y tres ranuras específicas para los LIDAR, lo que asegura una orientación precisa y funcional. El sistema de cierre se resuelve mediante anclajes en el borde del cuerpo y orejas externas para tornillos, lo que garantiza un ajuste firme y facilita el mantenimiento. A nivel general, el prototipo busca maximizar el aprovechamiento del espacio interno, mejorar la disposición de los sensores y proteger los componentes electrónicos. Es importante remarcar que se trata de un prototipo en evolución, sujeto a mejoras tanto en la disposición interna como en la ergonomía y robustez del ensamble.

5.2. Comparaciones entre Versiones

- **Espacio interno:** El primer prototipo no contaba con el espacio suficiente para albergar todos los componentes electrónicos. En el segundo, se corrigieron dimensiones y proporciones, logrando una organización más adecuada del volumen interno.
- **Ranuras para sensores LIDAR:** En el primer modelo, las ranuras eran más genéricas y no contemplaban bien la orientación de los sensores. El segundo prototipo introdujo tres ranuras específicas para los LIDAR, con un frente inclinado que mejora el ángulo de detección.
- **Relieves y fijaciones internas:** El primer y segundo prototipo carecían de soporte interno para los componentes. En el tercer prototipo, que exteriormente mantiene la misma geometría, se añadieron relieves interiores para que los LIDAR queden bien posicionados y se incorporaron finalmente los anclajes para fijar las plaquetas electrónicas, permitiendo un montaje más firme y seguro.
- Sistema de cierre: Desde el segundo prototipo ya se contemplaban orejas externas para tornillería, pero en el prototipos siguiente se mejoró la definición de los encastres en el borde del cuerpo, logrando un cierre más sólido.

- La serie de prototipos muestra una mejora constante en la funcionalidad y organización interna de la carcasa. El primer prototipo permitió validar dimensiones generales. El segundo resolvió la correcta orientación de los LIDAR con el frente inclinado y el aumento de espacio. El tercero introdujo finalmente los relieves interiores y fijaciones internas, mejorando la integración de los componentes. Este proceso iterativo confirma que el diseño está en el camino correcto, pero aún con margen para seguir perfeccionando aspectos de robustez estructural, ensamblaje y usabilidad.

5.3. Imágenes

5.3.1. Primer Prototipo

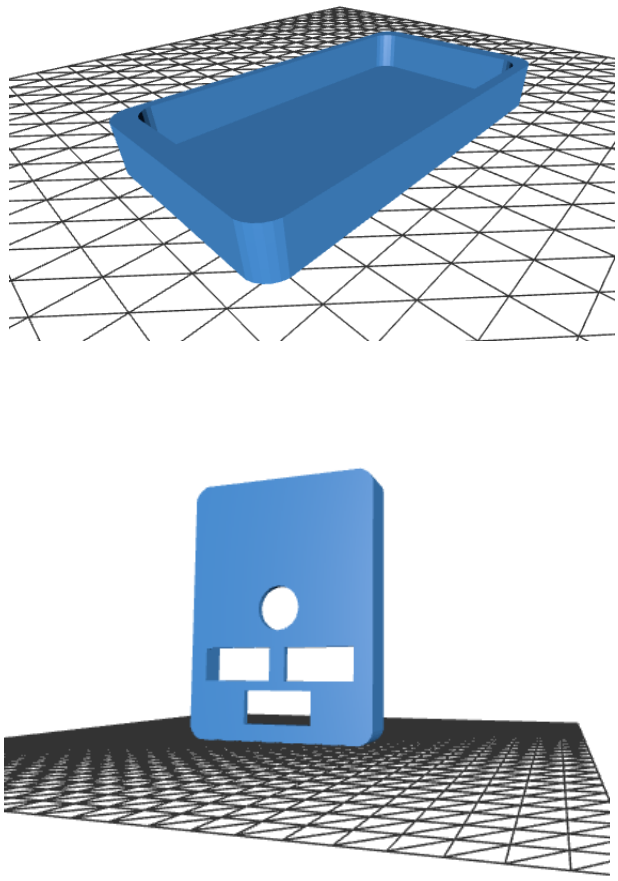


Figura 5.1: Primera tapa

5.3.2. Segundo Prototipo

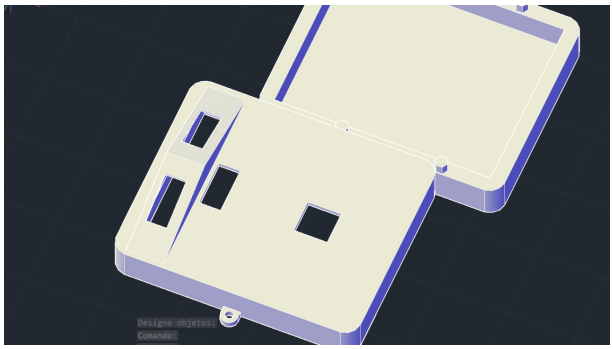


Figura 5.2: Tapa y base

Redes sociales

6.1. Página web

6.2. Lenguajes utilizados.

- El proyecto utiliza tres lenguajes principales: HTML, CSS y JavaScript. HTML define la estructura de la página, es decir, organiza el contenido en secciones, títulos, párrafos e imágenes. CSS se encarga de la parte visual, dándole color, tipografía, tamaños, efectos y adaptando el diseño a distintos dispositivos. Por último, JavaScript agrega la interactividad, como las animaciones, el carrusel de imágenes, los gráficos y la lectura en voz alta. En conjunto, HTML estructura, CSS diseña y JavaScript da dinamismo.

6.3. Desglose de Infraestructura

Header/Encabezado

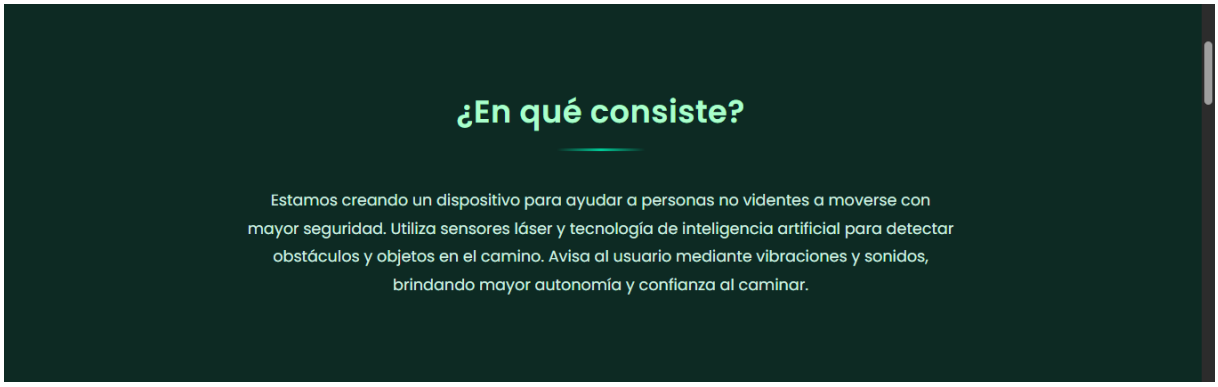
- Contenidos: Logo del proyecto, Nombre del proyecto, Slogan, Botón de lectura de contenidos por el narrador, Barra de navegación con todas las secciones de la web.
- Objetivo: Navegación cómoda, intuitiva y accesibilidad para el grupo objetivo.

Sección: ¿Quienes Somos?

- Contenidos: Perfiles de los integrantes del proyecto en recuadros por separado.
- Objetivos: Presentación.

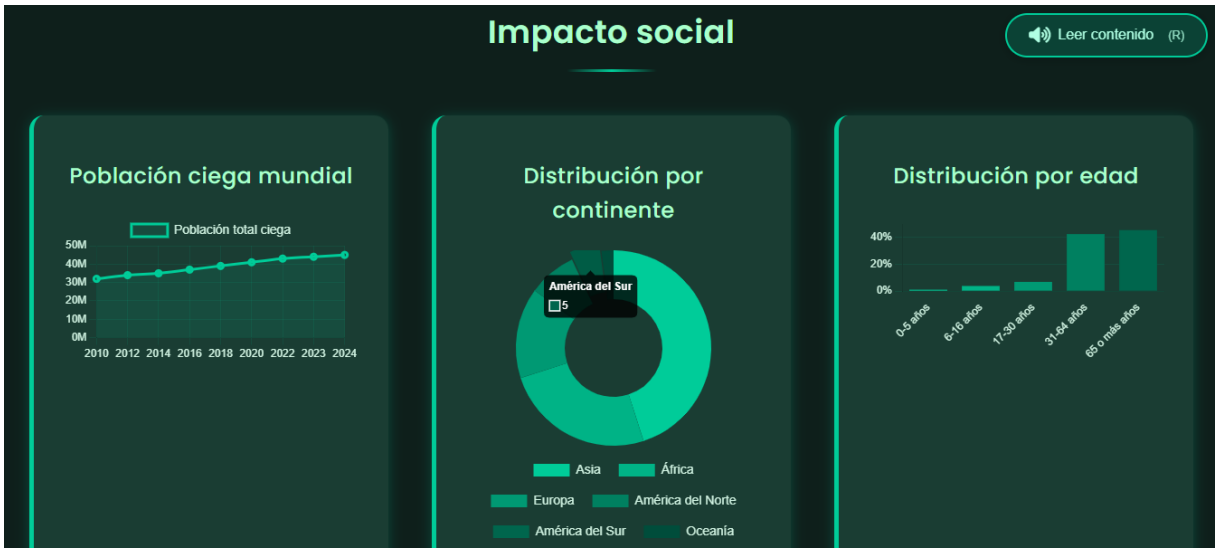
Sección: ¿En qué consiste?

- Contenidos: Texto.
- Objetivos: Explicar de manera muy breve el propósito del proyecto.



Sección: Impacto Social

- Contenidos: Gráficos: Población Ciega Mundial, Distribución por Continente, Distribución por Edad
- Objetivos: Concienciar sobre las estadísticas.

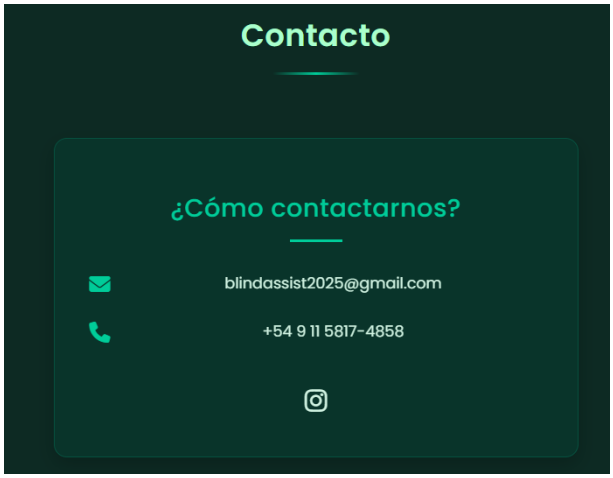


Sección: Detalles Técnicos

- Contenido: Cuadro de Texto.
- Objetivo: Breve explicación enfocada en el funcionamiento de los componentes.

Sección: Contacto

- Contenidos: Cuadro de contactos, Link a Instagram.
- Objetivo: Comunicarse con nostros.



Sección: Galería

- Contenido: Imágenes de presentaciones, proceso, etc.

6.4. Instagram

6.4.1. Introducción

El proyecto BlindAssist cuenta con un perfil en la red social Instagram (@blindassist), utilizado como canal principal de difusión y visibilización. La estrategia digital busca transmitir de manera clara y atractiva los valores del proyecto: innovación, accesibilidad y seguridad.

6.4.2. Colores Corporativos

Se emplea una paleta tecnológica y moderna compuesta por los tonos:

- #1B9E96
- #002C2D
- #004F4E

Estos colores transmiten confianza, innovación y un enfoque en la accesibilidad.

6.4.3. Logotipo

El isotipo está conformado por un ojo integrado con circuitos electrónicos, representando la unión entre la visión y la tecnología. El logotipo incluye el nombre BLINDASSIST en una tipografía moderna y clara, pensada para garantizar legibilidad.

6.4.4. Coherencia Gráfica

La identidad visual mantiene unidad en todas las piezas gráficas, incluyendo presentaciones, banners y publicaciones digitales, reforzando el reconocimiento de marca.

6.4.5. Formato y Planificación de Publicaciones

- El feed combina publicaciones gráficas y material en video corto (*reels*).
- Se utilizan banners con información incorporada en algunas imágenes para dar contexto y aportar valor visual a la grilla.
- El contenido está planificado de forma ordenada, priorizando la claridad estética y el atractivo visual.

6.4.6. Reels Publicados

- **Presentación del sponsor:** se destaca el apoyo recibido para la continuidad del proyecto.
- **Introducción general al proyecto:** explica el objetivo y la propuesta de BlindAssist.
- **Recepción de componentes:** muestra el avance del desarrollo a través de la llegada de insumos tecnológicos.

6.4.7. Objetivo del Perfil

El principal objetivo del Instagram es difundir el proyecto, acercándolo a la comunidad educativa, potenciales colaboradores y al público general. En esta etapa no se contemplan testimonios de usuarios finales, priorizando la visibilización institucional y técnica.

Referencias

6.5. Datasheets y manuales

[1] *Raspberry pi 4 datasheet:*. https://github.com/impatrq/BLINDSSIST/raspberry-pi-4-datasheet.pdf

[2] Raspberry Pi Foundation. *Raspberry Pi Camera Module v1.3 Documentation:*. https://www.raspberrypi.com/documentation/accessories/camera.html

[3] Benewake. *TFmini LiDAR Product Manual:* https://github.com/impatrq/BLINDSSIST/TFmini

[4] *TFmini Lidar Datasheet:* https://github.com/impatrq/BLINDSSIST/SJ-GU-TFmini-T-01$_A$05

[5] *Conversor serie a usb datasheet:* https://www.alldatasheet.com/datasheet-pdf/pdf/201067/SILABS/CP2102.html

[6] *Conversor buck step-down datasheet:* https://www.alldatasheet.com/datasheet-pdf/pdf/1134361/XLSEMI/XL4015.html

6.6. Software

[7] *Lectura de sensor TFmini:* https://github.com/impatrq/BLINDSSIST/TFmini3.py

[8] *Control de motores:* https://github.com/impatrq/BLINDSSIST/ContPWM.py

6.7. Redes sociales

[9] *Carpeta de Web BlindAssist* https://github.com/impatrq/BLINDSSIST/pagina

[10] *linktree de BlindAssist:* https://linktr.ee/blindassist

[11] *Pagina web de BlindAssist:* https://castiilloramiro.github.io/BlindWeb/

[12] *Instagram oficial de BlindAssist:* https://www.instagram.com/blindassist/

6.8. Documentación

[13] *Codigo en Tex de carpeta tecnica:* https://github.com/impatrq/BLINDSSIST/Codigo%20tex