Instituto Tecnológico Superior del Occidente del Estado de Hidalgo

División de Ingeniería en Tecnologías de la Información y Comunicaciones

REPORTE TECNICO

ESP32 – CAM PRODUCTO FINAL

RECONOCIMIENTO DE OBJETOS

1A

2025 – 2026

MTRO. Saúl Isaí Soto Ortiz

Luis Osvaldo Rufino Velázquez

Oscar Jesús Pérez Ramírez

Juan David Valera Montesinos

Mixquiahuala de Juárez, Hidalgo

19 de noviembre de 2025

Tabla de Contenidos

RESUMEN

Este reporte justifica la implementación de sistemas de visión artificial de bajo costo mediante *Edge Computing*. El objetivo fue desarrollar un prototipo con el módulo ESP32-CAM capaz de detectar y diferenciar un encendedor y una pinza utilizando la plataforma Edge-Impulse. Los resultados demostraron una identificación exitosa superior al 95%, comunicada visualmente mediante una pantalla OLED y LEDs, y auditivamente con un buzzer. Se concluye que el ESP32-CAM es viable y eficiente para ejecutar modelos de aprendizaje automático en el dispositivo.

INTRODUCCIÓN

Este informe justifica la necesidad de sistemas de reconocimiento visual de bajo costo. El objetivo fue desarrollar un prototipo con ESP32-CAM capaz de detectar y diferenciar dos objetos: un **Encendedor** y una **Pinza**. Como principales resultados, el sistema logró identificar ambos elementos exitosamente, mostrando el objeto detectado en una pantalla OLED. Además, activa un LED específico para cada objeto y emite una alerta sonora mediante un buzzer al confirmar la detección. Se concluye que el ESP32-CAM es viable para tareas de visión artificial simples con múltiples salidas.

El Reconocimiento de Objetos es una rama clave de la visión artificial. Su objetivo es identificar y clasificar elementos en una imagen o video, tarea que se realiza cada vez más con modelos de aprendizaje automático (Machine Learning) entrenados para reconocer patrones visuales. Históricamente, estas tareas requerían gran poder computacional. Sin embargo, la microelectrónica ha democratizado su acceso. El módulo ESP32-CAM es un ejemplo clave: es un microcontrolador de bajo costo que integra un procesador, conectividad Wi-Fi y una interfaz de cámara. Esta combinación lo hace ideal para el "Edge Computing", permitiendo procesar imágenes directamente en el dispositivo. El "qué" de este informe es la unión de estos conceptos. Se presenta un prototipo basado en el ESP32-CAM para demostrar su viabilidad en una tarea específica: la detección y diferenciación de dos objetos, un **Encendedor** y una **Pinza**. El sistema identifica el objeto y comunica el resultado al usuario mediante una pantalla OLED, indicadores LED diferenciados y una alerta sonora (buzzer). Las siguientes secciones servirán como guía del proyecto, detallando la metodología, la configuración del hardware y software, los resultados de las pruebas y las conclusiones sobre el rendimiento del sistema y su trabajo futuro.

MATERIALES

Para el proceso físico se usó:

- Esp32-Cam
- Modulo Esp32-Cam
- Encendedor
- Pinza
- Jumpers
- Leds
- Buzzer
- Protoboard
- Cautín
- Pasta para soldar
- Soldadura
- Base lonchbox

Para el proceso digital se usó:

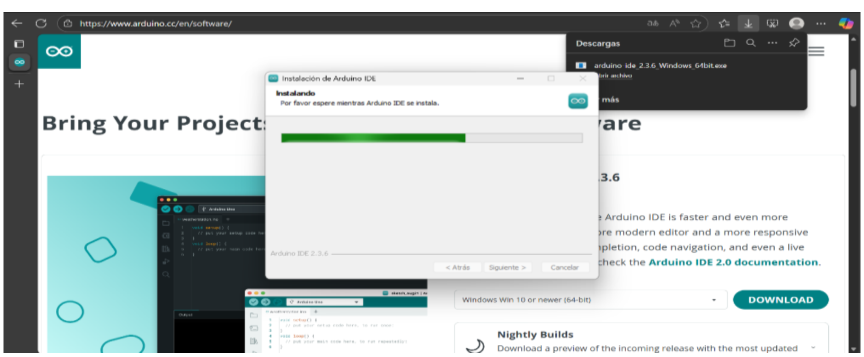
- Arduino IDE
- Librerías
- Edge-Impulse
- SolidWorks

PROCEDIMIENTO

El primer paso es configurar el entorno de desarrollo de Arduino IDE en Windows para que reconozca y pueda programarla placa ESP32-CAM.

3.1 Instalación de Arduino IDE:

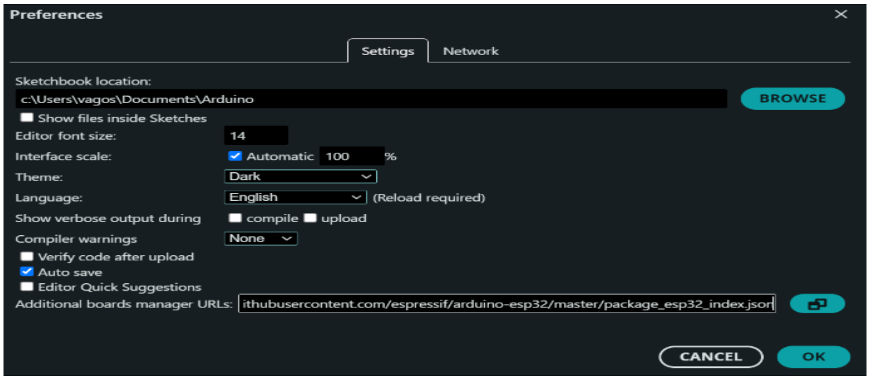
- Descargare instalar la última versión de Arduino IDE (versión 1.8.x o 2.x) desde el sitio web oficial.



3.2 Añadir el Gestor de Placas ESP32:

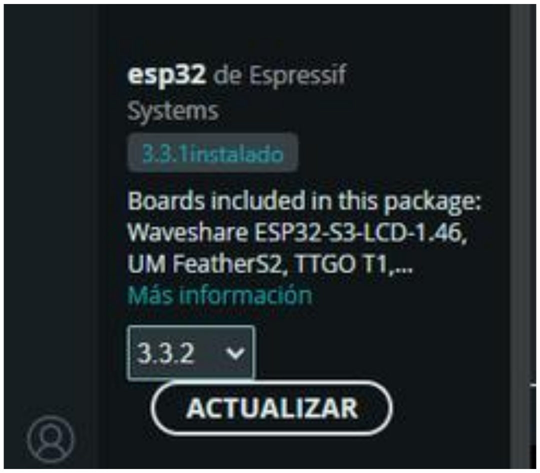
- Abrir Archivo > Preferencias.

- En "Gestor de URLs Adicionales de Tarjetas", añadir la siguiente URL:https://raw.githubusercontent.com/espressif/arduino-esp32/master/package_esp32_index.json



3.3 Instalar las Placas ESP32:

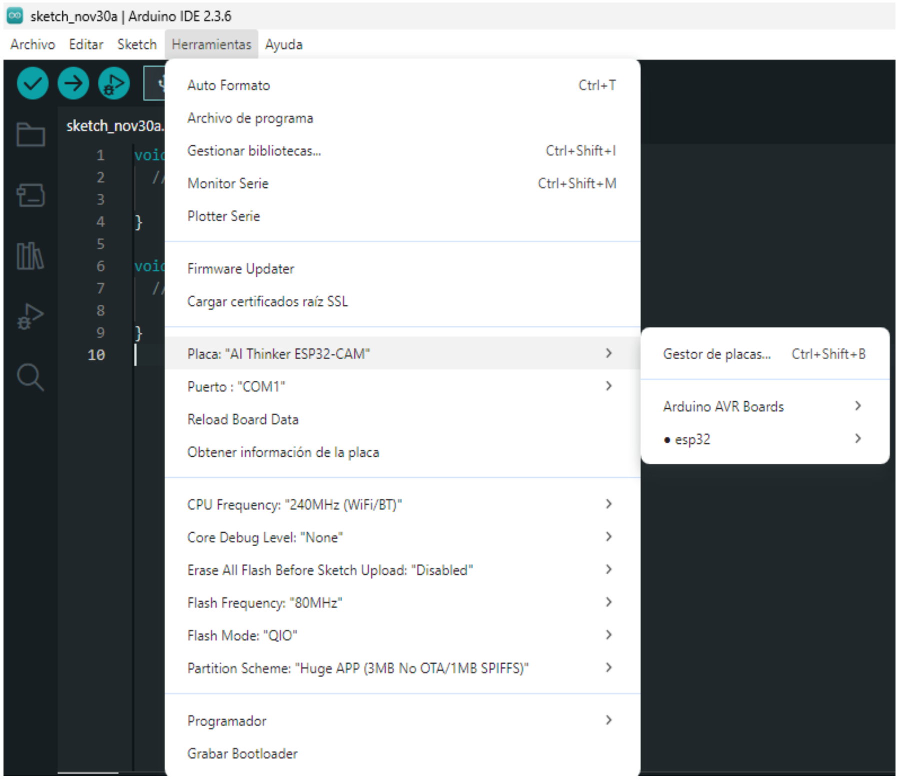
- Abrir Herramientas > Placa >Gestor de Tarjetas... Buscar "esp32" e instalar el paquete "esp32" by EspressifSystems".4



3.4 Configuración de la Placa en Arduino IDE:

- Conectar el ESP32-CAM a la PC mediante un programador USB-a-TTL (como el FTDI232).

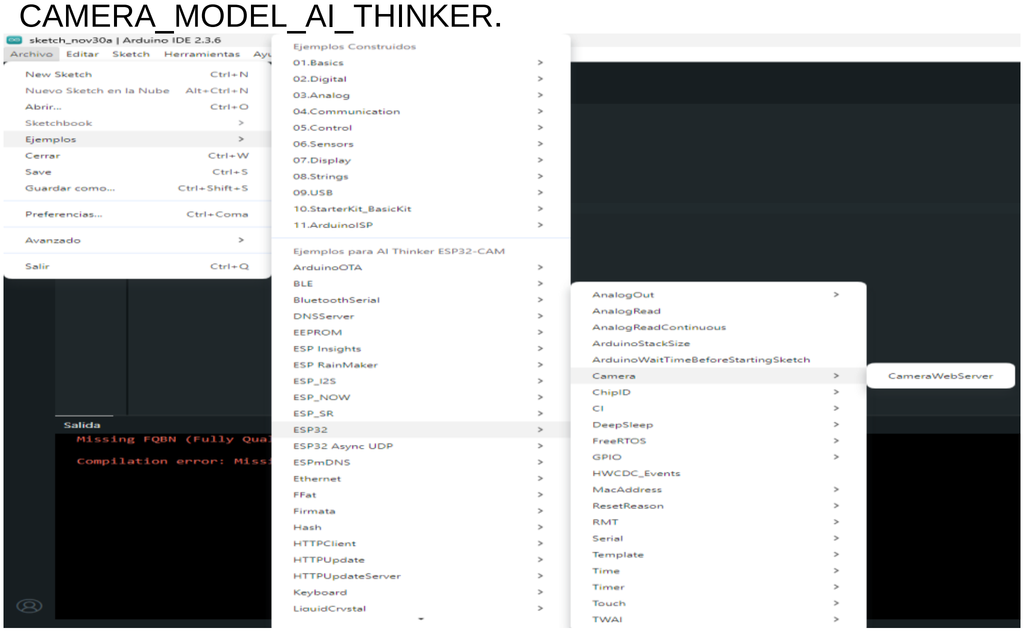
- En Herramientas > Placa, seleccionar "ESP32 WroverModule" (o "AI Thinker ESP32-CAM" si aparece).



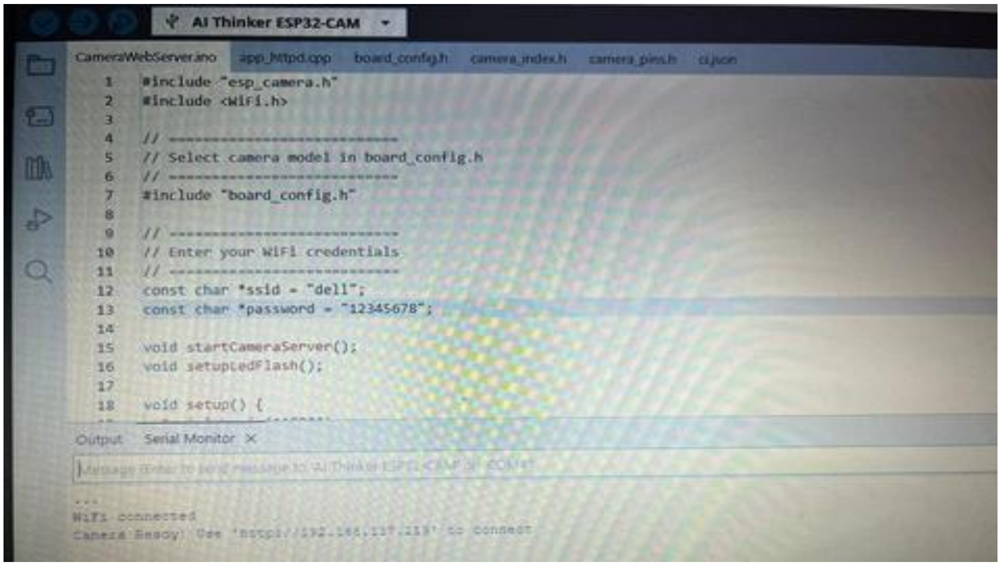
- Seleccionar el puerto COM correcto.

3.5 Prueba de Verificación (CameraWebServer):

- Abrir Archivo > Ejemplos > ESP32> Camera > CameraWebServer.Descomentar el modelo de placa CAMERA_MODEL_AI_THINKER.



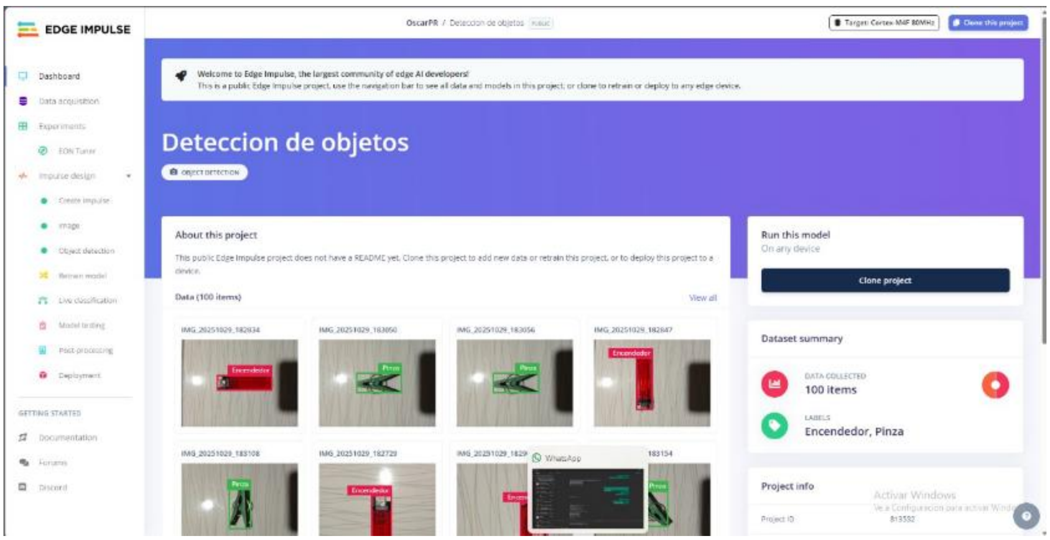
- Introducir las credenciales (SSID y contraseña) de la red Wi-Fi.
- Quitar el pin GPIO0 de GND y presionar el botón de Reset (RST) en la placa.
- Abrir el Monitor Serial (a 115200baudios) para ver la dirección IP asignada.



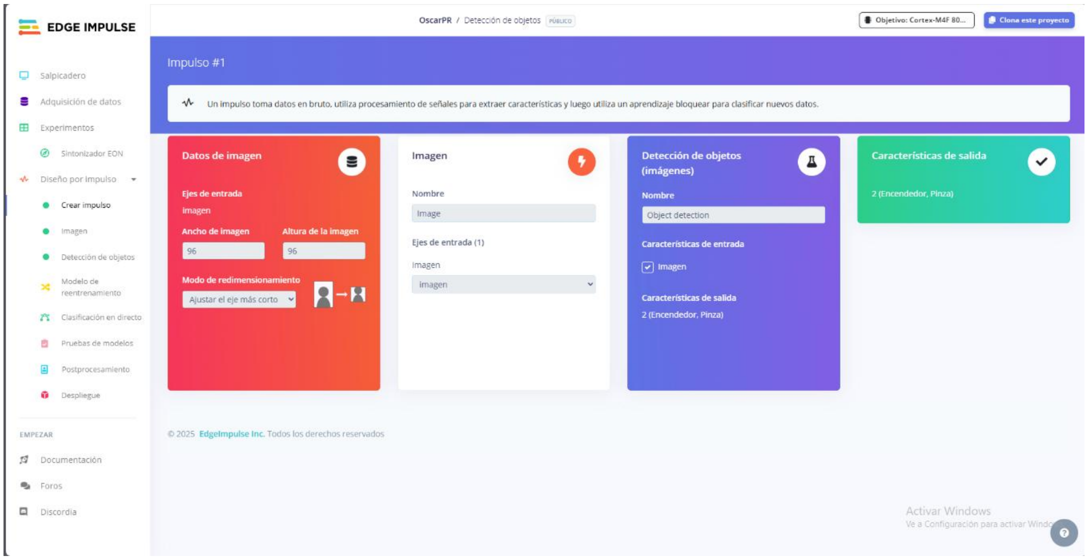
- Ingresar la IP en un navegador para verificar el streaming de video.

3.6 Edge-Impulse:

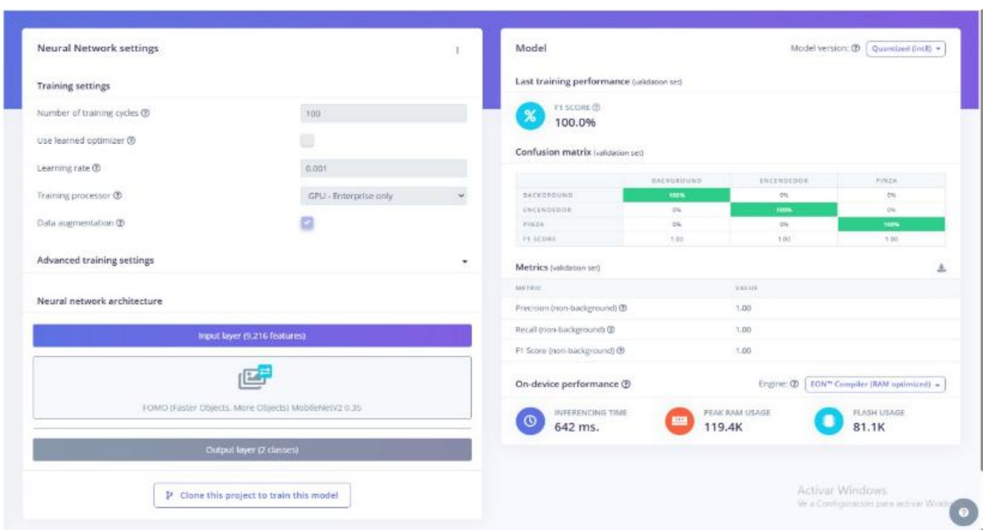
- Adquisición de Datos: Se recopilaron100-150 fotos por clase usando un móvil.



- Diseño del Impulso: Se configuró un pipeline optimizado para baja potencia, estableciendo las imágenes a 96x96 píxeles en color.

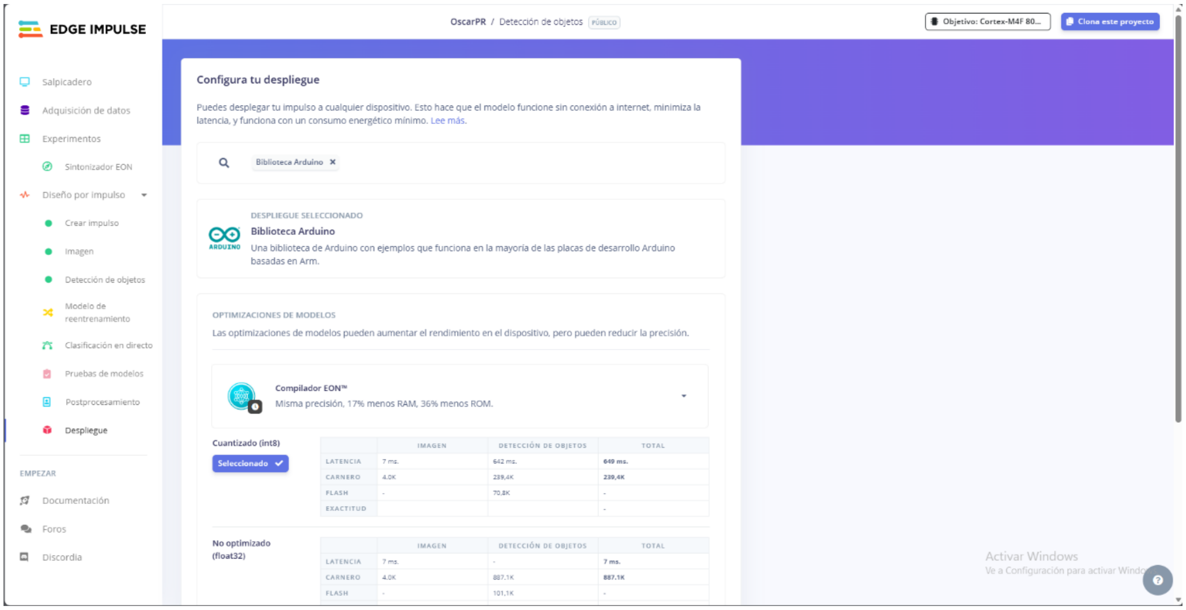


- Aprendizaje: Se empleó Transfer Learning (usando MobileNetV2 0.35) como bloque de aprendizaje.



- Entrenamiento: El modelo se entrenó ajustando parámetros hasta lograr una precisión superior al 95%.

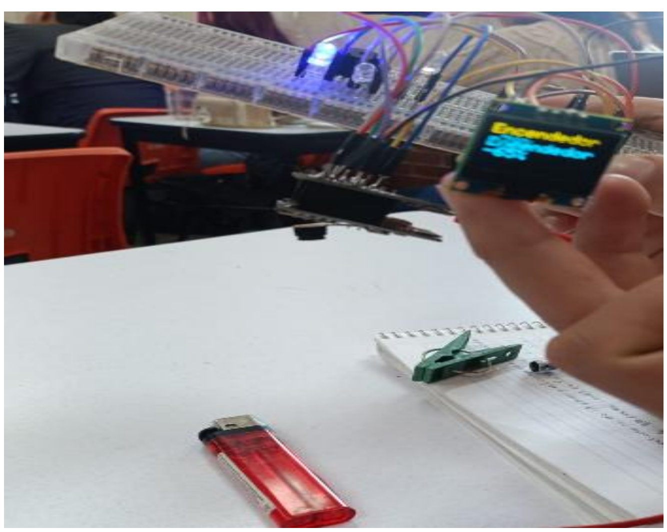
- Despliegue: Finalmente, el modelo completo se exportó como una Librería de Arduino (archivo.zip) lista para ser integrada en un microcontrolador.



3.7 Visualizar etiquetas:

- Monitor Serial: Sirve para depuración que confirma que la cámara y el modelo se iniciaron correctamente. En cada ciclo, imprime el tiempo que tardó la inferencia (ej. 250ms) y los puntajes de confianza para todas las clases (ej. Pinza: 0.85).

- Pantalla OLED: Una pantalla SSD1306 (I2C conectada a GPIO 21 y22) muestra la etiqueta del objeto con mayor confianza (ej. "OBJETO: PINZA"), pero solo si esa confianza supera un umbral del 80%. Si ningún objeto supera el umbral, muestra"...Buscando...".



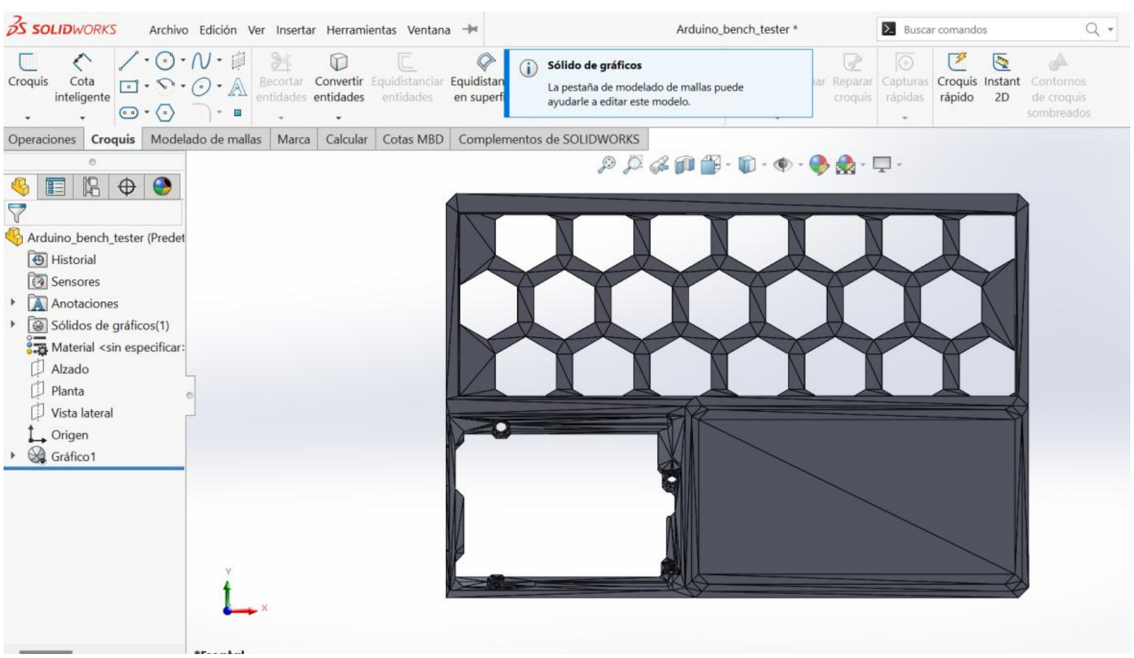
- Alertas Físicas (LEDs y Buzzer):
  - Un LED Rojo (GPIO 12) se enciende si detecta "Pinza".
  - Un LED Verde (GPIO 13) se enciende si detecta "Encendedor".
  - Si detecta "fondo" o la confianza es baja, ambos LEDs se apagan.
  - Un Buzzer (GPIO 14) emite un sonido corto como alerta auditiva cada vez que se identifica "Pinza" o "Encendedor".

RESULTADOS

El prototipo funcional del laboratorio utiliza una ESP32-CAM (modelo AI Thinker) para la detección de dos objetos específicos: una pila y un LED.
Los resultados de la detección se manifiestan activando simultáneamente un LED indicador (conectado al GPIO 12), un buzzer activo (conectado al GPIO 13) y mostrando un mensaje de confirmación en una pantalla OLED SSD1306 (conectada vía I2C a los pines GPIO 15 y GPIO 14).
La configuración del software se realizó en el entorno de Arduino IDE, requiriendo las librerías esp_camera.h para el sensor de imagen y Adafruit_SSD1306.h junto con Adafruit_GFX.h para la pantalla.

DESARROLLO DE BASE LONCHBOX

Para este usamos como referencia un lonchera o base tipo banco de trabajo, usamos SolidWorks para su creación, de mano de terceros imprimimos en un color negro mate, nuestra lonchbox para nuestro prototipo.



CONCLUSIONES

1. **Luis Osvaldo Rufino Velázquez:** Lo más relevante fue validar la capacidad del **ESP32-CAM** para realizar inferencia de *Machine Learning* sin depender de la nube. A través del uso de *Transfer Learning* con MobileNetV2, se logró optimizar el procesamiento de imágenes (96x96 píxeles) para obtener una precisión alta (>95%) en la diferenciación de objetos (encendedor vs. pinza), demostrando que la visión artificial es accesible en microcontroladores económicos.

2. **Oscar Jesús Pérez Ramírez:** Destaca la integración exitosa de múltiples periféricos de salida para mejorar la interfaz usuario-máquina. La configuración de pines GPIO para manejar simultáneamente la comunicación I2C con la pantalla **OLED SSD1306** y las señales digitales para los LEDs y el buzzer fue crítica. Esto permitió que el sistema no solo "viera" los objetos, sino que retroalimentara al usuario en tiempo real con etiquetas de confianza y alertas sonoras.

3. **Juan David Valera Montesinos:** El desarrollo del prototipo físico, incluyendo la base "Lonchbox" diseñada en **SolidWorks**, fue fundamental para la estabilidad de las pruebas. Se concluye que para una detección fiable no basta con el código; la consistencia en la captura de las 100-150 fotos por clase y la estabilidad mecánica del sensor son vitales para evitar falsos positivos y asegurar que el modelo entrenado en Edge-Impulse funcione correctamente en el entorno real.

REFERENCIAS

1. Edge Impulse, "Object Detection with FOMO (Faster Objects, More Objects)," *Edge Impulse Documentation*, 2024. [En línea]. Disponible: https://docs.edgeimpulse.com/docs/edge-impulse-studio/learning-blocks/object-detection/fomo-object-detection-for-constrained-devices.
2. Espressif Systems, "ESP32-CAM Especificaciones Técnicas y Hoja de Datos," *Espressif Documentation*, 2023. [En línea]. Disponible: https://www.espressif.com/en/products/socs/esp32.
3. L. Llamas, "ESP32-CAM: Qué es, pinout y cómo programarla con Arduino IDE," *LuisLlamas.es*, 20 de agosto de 2020. [En línea]. Disponible: https://www.luisllamas.es/esp32-cam-pinout-arduino-ide/.
4. Prometec, "Manejo de displays OLED I2C con controladores SSD1306," *Prometec.net*, 2022. [En línea]. Disponible: https://www.prometec.net/displays-oled-i2c/.
5. Gómez, "Introducción al aprendizaje profundo y visión por computador en sistemas embebidos," *Revista de Tecnología e Innovación*, vol. 5, no. 2, pp. 45-52, 2021.