VISION ARTIFICIAL

Detección de Botellas en Tiempo Real con Visión Artificial y Activado de Led en ESP32

NOMBRE: Leonardo Isaac Gonzalez Yañez

15/06/2025

Detección de Botellas en Tiempo Real con Visión Artificial y Activado de Led en ESP32

OBJETIVO

Implementar un sistema de visión artificial en Python capaz de detectar botellas en vivo utilizando la cámara de la computadora, y que, al encontrar una botella en la escena, envíe una señal Serial al ESP32 para que encienda o apague un led en consecuencia.

Objetivos específicos:

- Utilizar el modelo YOLOv8 para detectar objetos en vivo con la cámara.
- Filtrar las detecciones específicamente para el objeto "botella" o "bottle" en el modelo COCO.
- Establecer una comunicación Serial con la placa ESP32.
- Encender o apagar un LED en el ESP32 en función de la detección en la cámara.

EXPLICACIÓN GENERAL

Este proyecto combina visión artificial, comunicación Serial y control de hardware. Usando el modelo de detección de objetos YOLOv8, podemos detectar en tiempo real determinados objetos en el vídeo de la cámara.
En el código de Python, cuando se descubre una botella en el rango de visión de la cámara, el computadora envía a través del puerto Serial un "1", y cuando deja de aparecer, envía un "0".
Este 1 o 0 es leído en el código de Arduino en el ESP32, que enciende o apaga el led según el valor recibido.

CÓDIGO (Python)

python

CopiarEditar

```
import cv2

from ultralytics import YOLO

import serial

import time
```

```
# Configurar puerto Serial
esp32 = serial.Serial('COM4', 115200)  # Ajusta según tu puerto
time.sleep(2)  # Espera a que el Serial esté listo

model = YOLO('yolov8n.pt')  # Modelo pequeño

cap = cv2.VideoCapture(0)

try:
    while True:
        ret, frame = cap.read()
        if not ret:
            break

        results = model(frame)
        result = results[0]

        botella_detectada = False

        for box, cls, score in zip(result.boxes.xyxy, result.boxes.cls, result.boxes.conf):
            class_name = model.names[int(cls)]

            if class_name == 'bottle' and float(score) > 0.5:
                botella_detectada = True
                x1, y1, x2, y2 = box.int().tolist()
```

```
            cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)
            cv2.putText(frame, f"Bottle {score:.2f}", (x1, y1-10),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.6, (0, 255, 0), 2)

        if botella_detectada:
            esp32.write(b'1')
        else:
            esp32.write(b'0')

        cv2.imshow("Detección Botella", frame)

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

finally:
    cap.release()
    esp32.close()
    cv2.destroyAllWindows()
```

CÓDIGO (ESP32)

cpp

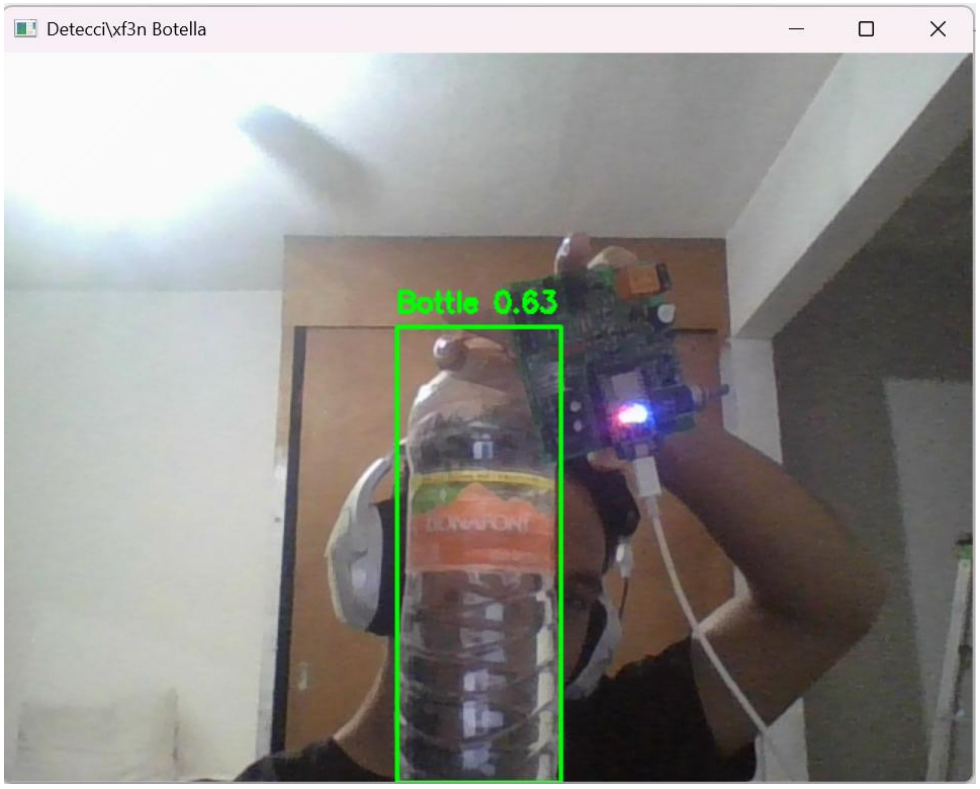
CopiarEditar

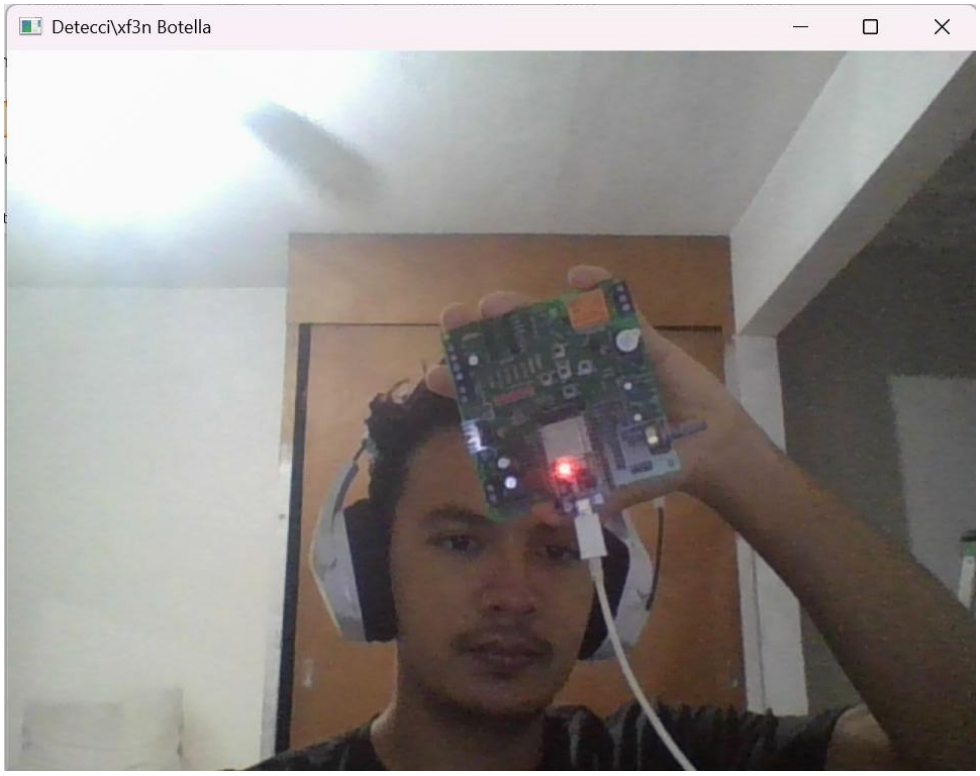
```
void setup()
{
  Serial.begin(115200);
  pinMode(2, OUTPUT);
```

```
}


void loop()
{
  if (Serial.available()) {
    char cmd = Serial.read();


    if (cmd == '1') { // Botella detectada
      digitalWrite(2, HIGH);
    } else if (cmd == '0') { // Botella NO detectada
      digitalWrite(2, LOW);
    }
  }
}
```

PRUEBAS

Durante las pruebas, el modelo de detección de objetos respondió de forma adecuada cuando aparecía una botella en la cámara.
La tasa de detección fue estable con una **confianza > 0.5**, evitando así falsos positivos.
La comunicación Serial estuvo estable, sin perder mensajes, y el led en el ESP32 respondió de forma instantánea cuando el objeto aparecía o desaparecía en el vídeo.


Detecci\xf3n Botella
Bottle 0.63


Detecci\xf3n Botella

ERRORES ENCONTRADOS

Durante las pruebas, primero la cámara no se estaba habilitando. Esto se resolvió instalando la librería opencv-python.

Posteriormente, el puerto Serial no estaba correcto (porque no estaba elegido en Arduino IDE o estaba ocupado), pero al verificar en administrador de dispositivos que puerto estaba utilizando el ESP32, se resolvió el problema.

Finalmente, cuando el modelo estaba muy exigido, la tasa de fotogramas caía. Esto se resolvió utilizando el modelo más liviano de Yolov8, yolov8n.pt.

🔹 CONCLUSIÓN

Este proyecto proporciona una demostración práctica de cómo aplicar la visión artificial en el control de dispositivos en el mundo real. Combinando Python, OpenCV, YOLOv8, Serial y ESP32, podemos detectar objetos específicos en un vídeo en vivo y ejecutar una acción física en consecuencia. Este procedimiento puede llevarse a muchos más casos, como domótica, robótica, seguridad, logística, etc., aumentando así el rango de aplicación de la Inteligencia Artificial en entornos industriales o de uso cotidiano.

Link video:

https://drive.google.com/drive/folders/1EkC7J5CnK8m-kCHRBUn1Ps49CTE1hMw-?usp=sharing

Link github:

https://github.com/Leo1Glez99/Visi-n-Artificial/tree/main/Proyecto%20deteccion%20de%20objetos