import cv2

import numpy as np

import RPi.GPIO as GPIO

import time

import pygame

import requests

# Initialize pygame mixer for playing sound

pygame.mixer.init()

# Sound file paths

BIRD\_SOUND\_1 = "/home/pi/opencv-venv/mixkit-horde-of-barking-dogs-60(1).wav"

BIRD\_SOUND\_2 = "/home/pi/opencv-venv/mixkit-8-bit-explosion-gun-2779.wav"

# YOLO Model Paths

YOLO\_CONFIG = "/home/pi/opencv-venv/yolov4.cfg"

YOLO\_WEIGHTS = "/home/pi/opencv-venv/yolov4.weights"

COCO\_NAMES = "/home/pi/opencv-venv/coco.names"

# Servo motor pin setup

SERVO\_PIN = 17 # The GPIO pin to which the servo is connected

# GPIO setup

GPIO.setmode(GPIO.BCM)

GPIO.setup(SERVO\_PIN, GPIO.OUT)

# Set up PWM for servo motor (50Hz)

servo = GPIO.PWM(SERVO\_PIN, 50) # 50Hz PWM frequency

servo.start(0) # Initialize with 0% duty cycle

# YOLO network setup

net = cv2.dnn.readNetFromDarknet(YOLO\_CONFIG, YOLO\_WEIGHTS)

layer\_names = net.getLayerNames()

output\_layers = [layer\_names[i - 1] for i in net.getUnconnectedOutLayers()]

# Load class names (COCO dataset)

with open(COCO\_NAMES, "r") as f:

classes = [line.strip() for line in f.readlines()]

cap = cv2.VideoCapture(0)

# Telegram Bot Configuration

BOT\_TOKEN = "7614924300:AAEfCPj3aQRJVlKcfuBL3XtcLhntQ1QK1ds"

CHAT\_ID = "6315300360"

MESSAGE = "Alert! A bird has been detected near your crops."

def send\_telegram\_notification():

"""Sends a Telegram alert when a bird is detected."""

url = f"https://api.telegram.org/bot{BOT\_TOKEN}/sendMessage"

data = {"chat\_id":CHAT\_ID, "text":MESSAGE}

requests.post(url, data=data)

# Servo motor control function

def rotate\_servo\_to\_angle(angle):

"""Rotates the servo motor to a specified angle."""

duty = angle / 18 + 2 # Calculate duty cycle (0 to 180 degrees)

GPIO.output(SERVO\_PIN, True)

servo.ChangeDutyCycle(duty)

time.sleep(1)

GPIO.output(SERVO\_PIN, False)

servo.ChangeDutyCycle(0) # Stop the PWM

try:

while True:

ret, frame = cap.read()

if not ret:

break

height, width, \_ = frame.shape

blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416), swapRB=True, crop=False)

net.setInput(blob)

detections = net.forward(output\_layers)

bird\_detected = False

for detection in detections:

for obj in detection:

scores = obj[5:]

class\_id = np.argmax(scores)

confidence = scores[class\_id]

detected\_class = classes[class\_id]

if confidence > 0.5 and detected\_class == "bird":

bird\_detected = True

print(f"Detected: {detected\_class} with confidence {confidence:.2f}")

# Actions when a bird is detected

if bird\_detected:

send\_telegram\_notification() # Send Telegram alert

pygame.mixer.music.load(BIRD\_SOUND\_1)

pygame.mixer.music.play()

time.sleep(2) # Wait for sound to play

# Rotate the servo motor

rotate\_servo\_to\_angle(0)

time.sleep(1)

rotate\_servo\_to\_angle(90)

time.sleep(1)

rotate\_servo\_to\_angle(180)

time.sleep(1)

rotate\_servo\_to\_angle(90)

time.sleep(1)

rotate\_servo\_to\_angle(0)

pygame.mixer.music.load(BIRD\_SOUND\_2)

pygame.mixer.music.play()

time.sleep(2) # Wait for sound to play

delay\_time = 30 # Bird detected ? 30 seconds delay

else:

delay\_time = 10 # No bird detected ? 10 seconds delay

cv2.imshow("Live Feed", frame)

key = cv2.waitKey(1)

if key == ord("q"):

break

time.sleep(delay\_time) # Apply delay before capturing next frame

finally:

cap.release()

cv2.destroyAllWindows()

servo.stop() # Stop PWM

GPIO.cleanup()