**Vision-Based Object Tracking with Obstacle Avoidance**

* **Key Functionalities:**

1. **Object Detection using TensorFlow Lite (TFLite) Model:**
   * Uses **MobileNet SSD v2 COCO** model for object detection.
   * Detects objects from a predefined list (arr\_valid\_objects).
   * Uses OpenCV to capture video and track detected objects.
2. **Ultrasonic Sensor for Obstacle Detection:**
   * Measures the distance of obstacles using an **HC-SR04** sensor.
   * **Prevents collision** by stopping the robot if an obstacle is too close.
   * Uses SAFE\_DISTANCE = 30cm and CRITICAL\_DISTANCE = 15cm as thresholds.
3. **Flask Web Interface for Video Streaming:**
   * Runs a Flask server (app = Flask(\_\_name\_\_)).
   * Streams live video feed using /video\_feed endpoint.
4. **Motor Control using PWM on GPIO Pins:**
   * Uses GPIO **pins 20 and 21** for motor speed control.
   * Adjusts speed dynamically based on distance to obstacles.
5. **Object Tracking Algorithm:**
   * Calculates x\_deviation and y\_deviation from the center of the frame.
   * Moves the robot **left/right** if deviation is in the x-axis.
   * Moves the robot **forward/backward** if deviation is in the y-axis.
   * Uses a **separate thread (Thread(target=move\_robot))** for movement.
6. **Speed Adjustment Logic:**
   * If the robot is **too close to an obstacle, it stops**.
   * If within a **safe range, speed is reduced dynamically**.

* **Possible Issues & Improvements**

1. **Timeout Handling in Ultrasonic Sensor:**
   * The function measure\_distance() includes timeouts, but excessive sensor noise can still cause errors.
   * Consider adding an **average filter (like a moving average) to smooth out sensor readings.**
2. **CPU Performance Optimization:**
   * cap = cv2.VideoCapture(0) runs on the main thread, which might block performance.
   * Solution: Move video capture and processing to a **separate thread** to prevent lag.
3. **Handling Cases When No Object is Found:**
   * The function track\_object(objs, labels) **stops the robot** when no object is detected.
   * But it **does not attempt to search for the object again**.
   * Solution: Implement a **scanning behavior (slow rotation) to re-acquire objects**.
4. **Speed Control Could Be Smoother:**
   * The speed changes suddenly in adjust\_speed\_by\_distance(distance).
   * Solution: Implement **gradual acceleration and deceleration** instead of direct speed jumps.