

#### **4. DESIGN AND TRAIN A MODEL FOR OBJECTS DETECTION WITH REAL TIME EXAMPLE**

<b>EX.N0 : 4</b>	<b>DESIGN AND TRAIN A MODEL FOR OBJECTS DETECTION WITH REAL TIME EXAMPLE</b>
<b><u>DATE : 20/02/2025</u></b>	

#### **AIM:**

To design and train a real-time object detection model using YOLO to detect objects such as cars, people, or other items in video or camera feed.

#### **ALGORITHM:**

Step 1: Import necessary libraries (TensorFlow, OpenCV, NumPy).

Step 2: Load a pre-trained YOLO model (YOLOv3, for example).

Step 3: Load and pre-process the input video or camera feed.

Step 4: Visualize detected objects with bounding boxes and class labels.

Step 5: Output the video with real-time object detection.

#### **PROGRAM:**

```
import cv2
import numpy as np
yolo_model = "yolov3.weights" # Path to YOLOv3 weights
yolo_cfg = "yolov3.cfg"      # Path to YOLOv3 configuration
yolo_names = "coco.names"    # File with class names (e.g., coco dataset)
net = cv2.dnn.readNet(yolo_model, yolo_cfg)
layer_names = net.getLayerNames()
output_layers = [layer_names[i - 1] for i in net.getUnconnectedOutLayers()]
with open(yolo_names, 'r') as f:
    classes = [line.strip() for line in f.readlines()]
```

```

cap = cv2.VideoCapture(0) # 0 for webcam, or video file path for custom video
while True:
    ret, frame = cap.read()
    if not ret:
        break
    blob = cv2.dnn.blobFromImage(frame, 0.00392, (416, 416), (0, 0, 0), True, crop=False)
    net.setInput(blob)
    outs = net.forward(output_layers)
    class_ids, confidences, boxes = [], [], []
    height, width, channels = frame.shape
    for out in outs:
        for detection in out:
            scores = detection[5:]
            class_id = np.argmax(scores)
            confidence = scores[class_id]
            if confidence > 0.5: # Set confidence threshold (e.g., 50%)
                center_x = int(detection[0] * width)
                center_y = int(detection[1] * height)
                w = int(detection[2] * width)
                h = int(detection[3] * height)
                x = center_x - w // 2
                y = center_y - h // 2
                boxes.append([x, y, w, h])
                confidences.append(float(confidence))
                class_ids.append(class_id)
    indexes = cv2.dnn.NMSBoxes(boxes, confidences, score_threshold=0.5, nms_threshold=0.4)
    for i in range(len(boxes)):
        if i in indexes:
            x, y, w, h = boxes[i]
            label = str(classes[class_ids[i]])
            confidence = str(round(confidences[i], 2))

```

```
color = (0, 255, 0) # Green color for boxes
cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)
cv2.putText(frame, label + " " + confidence, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
color, 2)
cv2.imshow('Object Detection', frame)
if cv2.waitKey(1) & 0xFF == ord('q'):
break
cap.release()
cv2.destroyAllWindows()
```

### **OUTPUT:**



### **RESULT:**

Thus the Program has been executed successfully and verified.