EX NO: 5	**Multiple Object Detection using YOLO**

Aim:

To detect and visualize objects in a specific frame of a video using the YOLOv8 model with PyTorch and OpenCV, and enhance the output with custom bounding boxes and labels.

Algorithm:

1. **Import Libraries:** Load required libraries such as `cv2`, `torch`, `ultralytics`, `matplotlib`, etc.
2. **Device Setup:** Choose GPU if available, else use CPU.
3. **Load YOLOv8 Model:** Load the lightweight `yolov8n.pt` model.
4. **Read Video Frame:** Access the specified frame (e.g., 150) from the video.
5. **Preprocess Frame:** Convert the frame from BGR to RGB for inference.
6. **Run Inference:** Detect objects using YOLOv8.

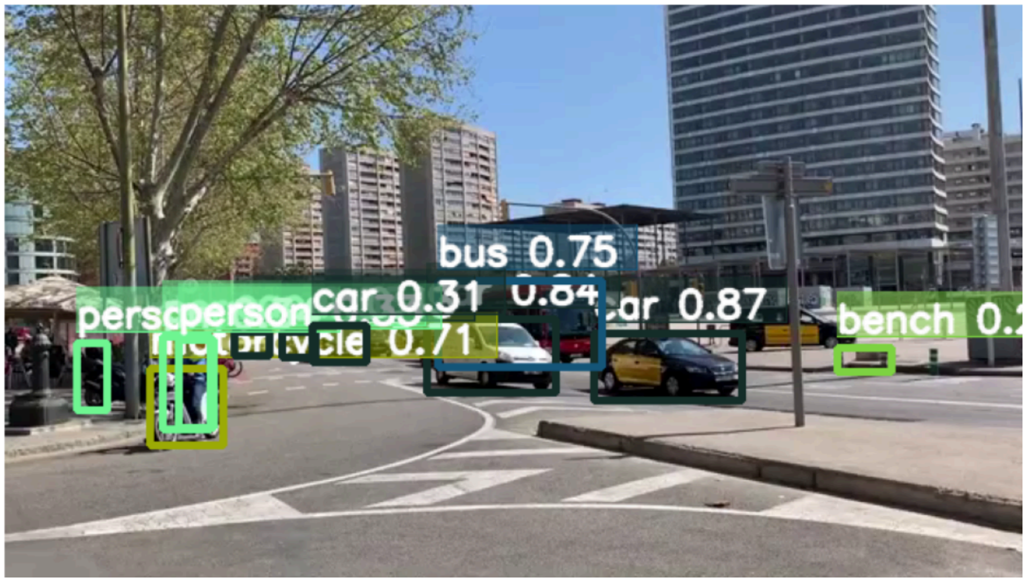
Code:

```
import cv2

import torch

import random

import numpy as np

import matplotlib.pyplot as plt
```

```
from ultralytics import YOLO

device = "cuda" if torch.cuda.is_available() else "cpu"

model = YOLO("yolov8n.pt").to(device)

video_path = "/content/road_trafifc (1).mp4"

cap = cv2.VideoCapture(video_path)

frame_number = 150

cap.set(cv2.CAP_PROP_POS_FRAMES, frame_number)

ret, frame = cap.read()

if not ret:

    print(f"Failed to read frame {frame_number}.")

    cap.release()

    exit()

frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

results = model(frame_rgb)

class_colors = {i: (random.randint(0, 255), random.randint(0, 255), random.randint(0, 255)) for i
in range(len(model.names))}

for result in results:

    for box in result.boxes:

        x1, y1, x2, y2 = map(int, box.xyxy[0])

        confidence = box.conf[0]

        class_id = int(box.cls[0])

        label = model.names[class_id]

        color = class_colors[class_id]
    cv2.rectangle(frame_rgb, (x1, y1), (x2, y2), color, 4)

            (text_width, text_height), _ = cv2.getTextSize(f"{label} {confidence:.2f}",
cv2.FONT_HERSHEY_SIMPLEX, 0.8, 2)

    text_offset_x, text_offset_y = x1, y1 - 12

     if frame_rgb[text_offset_y - text_height - 5:text_offset_y + 5, text_offset_x:text_offset_x +
text_width + 10].shape[0] > 0:

              overlay = frame_rgb[text_offset_y - text_height - 5:text_offset_y + 5,
text_offset_x:text_offset_x + text_width + 10].copy()

        cv2.rectangle(frame_rgb, (text_offset_x, text_offset_y - text_height - 5), (text_offset_x +
text_width + 10, text_offset_y + 5), color, -1)

         cv2.addWeighted(overlay, 0.5, frame_rgb[text_offset_y - text_height - 5:text_offset_y +
5, text_offset_x:text_offset_x + text_width + 10], 0.5, 0, frame_rgb[text_offset_y - text_height -
5:text_offset_y + 5, text_offset_x:text_offset_x + text_width + 10])

              cv2.putText(frame_rgb, f"{label} {confidence:.2f}", (x1, y1 - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255, 255, 255), 2, cv2.LINE_AA)


plt.figure(figsize=(12, 7))

plt.imshow(frame_rgb)

plt.axis("off")

plt.title(f"Frame {frame_number} - YOLO Detection")

plt.show()
```

Output:

Frame 150 - YOLO Detection



Result:

The YOLOv8 model accurately identified multiple objects (e.g., vehicles, people) in the selected video frame. Each detection is shown with a colored bounding box and a readable label with confidence. The annotated frame is displayed clearly using `matplotlib`.