Implement Video Analytics for a Live Video

Aim:

To implement video analytics techniques on a live video stream for object detection, motion detection, and event analysis.

Objective:

- To capture live video from webcam/CCTV
- To detect moving objects
- To perform real-time object detection
- To analyze video frames for forensic purposes

Theory:

Video Analytics ante live or recorded video ni analyze chesi meaningful information extract cheyyadam.

Cybercrime & Digital Forensics lo video analytics use cases:

- CCTV footage analysis
- Face detection
- Object tracking
- Suspicious activity detection
- Crowd monitoring

Live video stream ni frames ga divide chesi, prathi frame meeda algorithm apply chestaru.

Types of Video Analytics

1 Motion Detection

Background subtraction technique use chestaru.

2 Object Detection

AI/ML models (YOLO, Haar Cascade, etc.) use chestaru.

3 Face Detection

Human faces identify cheyyadam.

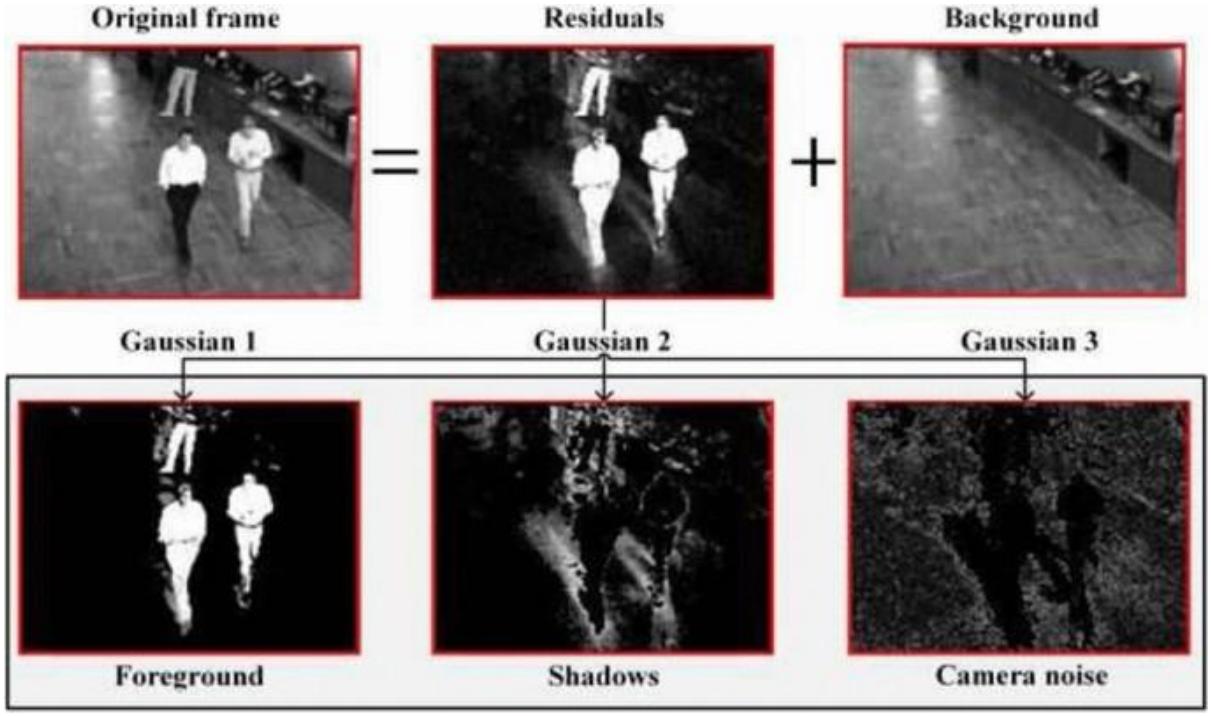
4 Object Tracking

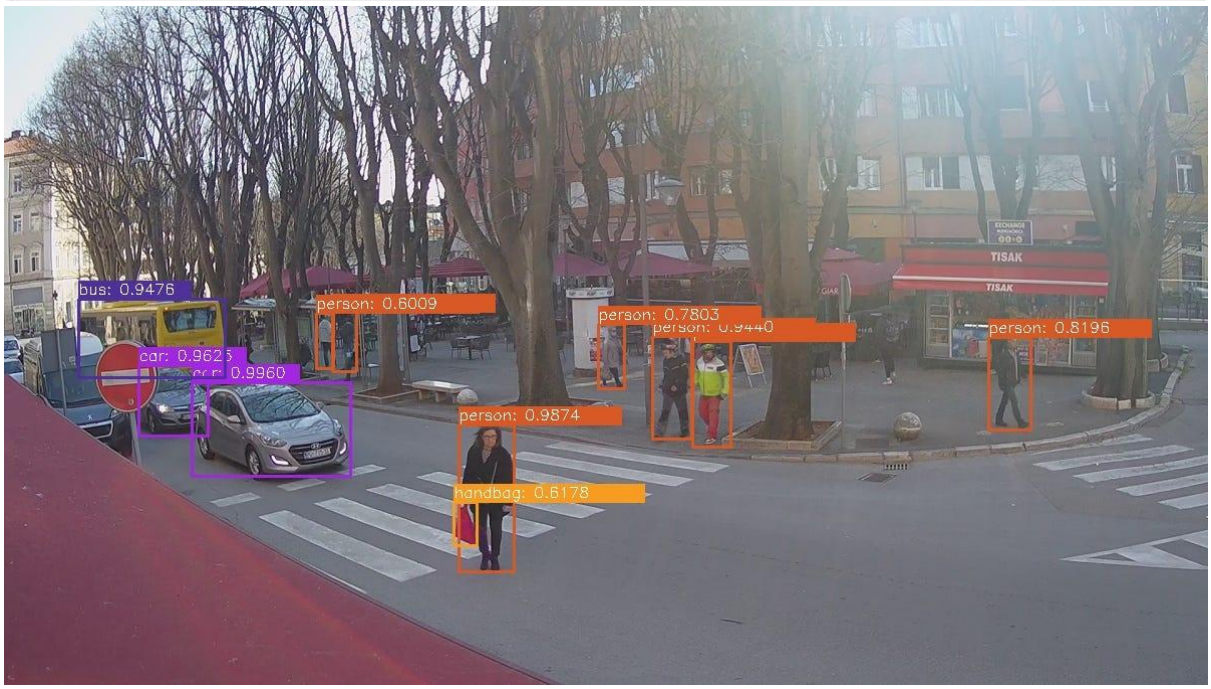
Moving object ni continuous ga track cheyyadam.

Tools & Technologies Used

- Python
- OpenCV
- Webcam / CCTV camera
- AI models (YOLO / Haarcascade)
- TensorFlow (optional)

Procedure: Motion Detection using OpenCV





Step 1 : Install Required Libraries

pip install opencv-python

Step2 : Capture Live Video

```
import cv2

cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    cv2.imshow("Live Video", frame)

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

cap.release()
cv2.destroyAllWindows()
```

Step3 : Implement Motion Detection

```
import cv2

cap = cv2.VideoCapture(0)
ret, frame1 = cap.read()
ret, frame2 = cap.read()

while cap.isOpened():
    diff = cv2.absdiff(frame1, frame2)
    gray = cv2.cvtColor(diff, cv2.COLOR_BGR2GRAY)
    blur = cv2.GaussianBlur(gray, (5,5), 0)
    _, thresh = cv2.threshold(blur, 20, 255, cv2.THRESH_BINARY)
    dilated = cv2.dilate(thresh, None, iterations=3)

    contours, _ = cv2.findContours(dilated, cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)

    for contour in contours:
        if cv2.contourArea(contour) < 1000:
            continue
        x, y, w, h = cv2.boundingRect(contour)
        cv2.rectangle(frame1, (x,y), (x+w,y+h), (0,255,0), 2)

    cv2.imshow("Motion Detection", frame1)
    frame1 = frame2
    ret, frame2 = cap.read()

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

```
        break

cap.release()
cv2.destroyAllWindows()
```

Observations:

- Live video successfully captured
- Moving objects detected
- Bounding boxes displayed
- Frames processed in real-time

Result:

Live video analytics successfully implemented.
Motion detection performed using frame difference method.

Applications in Digital Forensics:

- CCTV evidence analysis
- Criminal movement tracking
- Vehicle detection
- Event reconstruction

Precautions:

- Ensure good lighting
- Stable camera position
- Proper system configuration
- Maintain video logs

Viva Questions:

1. What is Video Analytics?
2. What is Background Subtraction?
3. Difference between Motion Detection and Object Detection?
4. What is Frame Rate?
5. What is Contour Detection?

----------END----------

Extra Work

AI-Based Object Detection using YOLO (Live Video Analytics)

Aim:
To implement real-time object detection on live video using YOLO algorithm.

Objective:

- To understand YOLO (You Only Look Once) algorithm
- To detect multiple objects in live video
- To draw bounding boxes with class labels
- To apply AI in video forensics

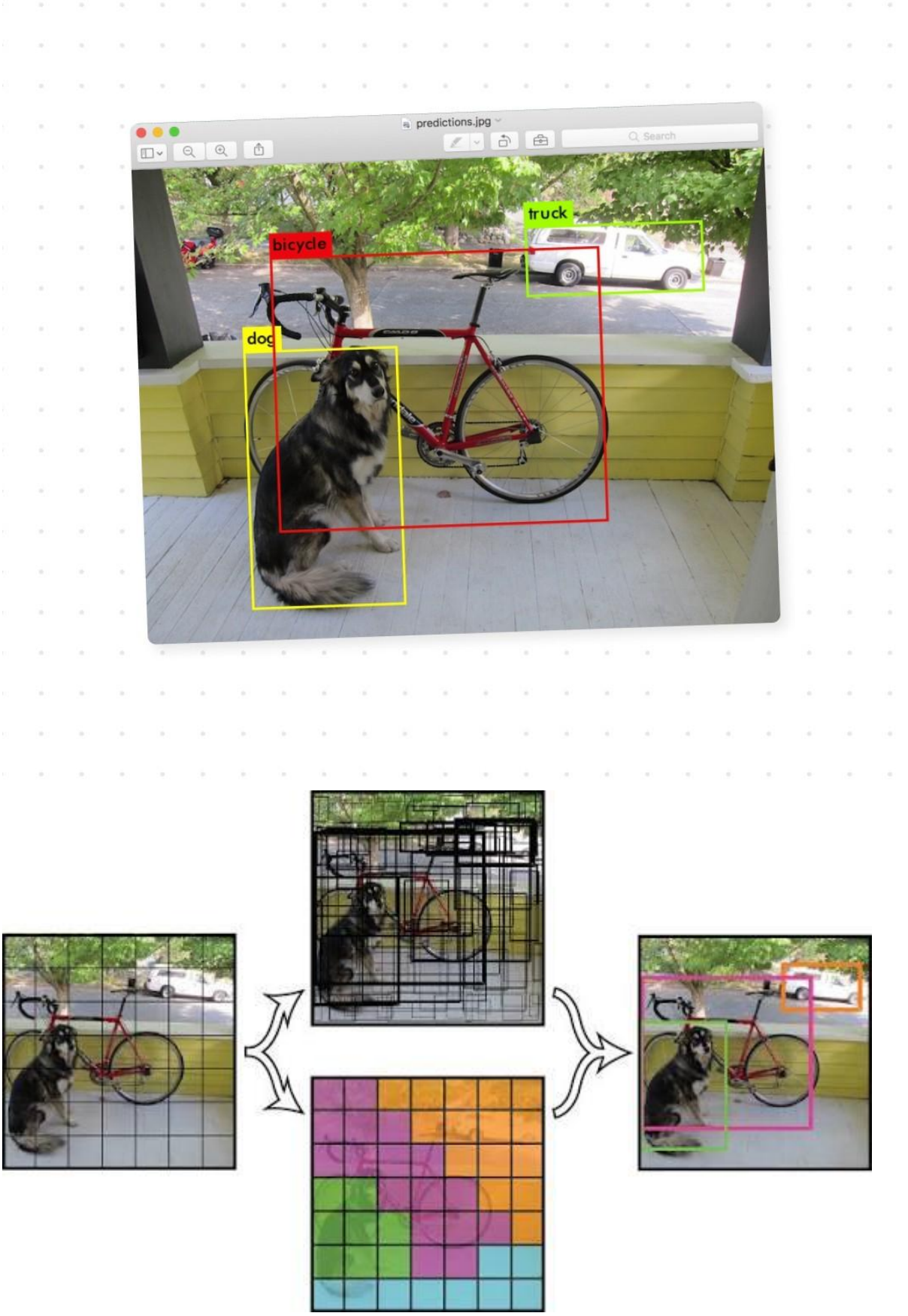
Theory:
YOLO (You Only Look Once) is a real-time object detection algorithm.
Single pass lo image ni process chesi multiple objects detect chestundi.
Traditional methods slow untayi.
YOLO fast & accurate – live CCTV analytics ki suitable.

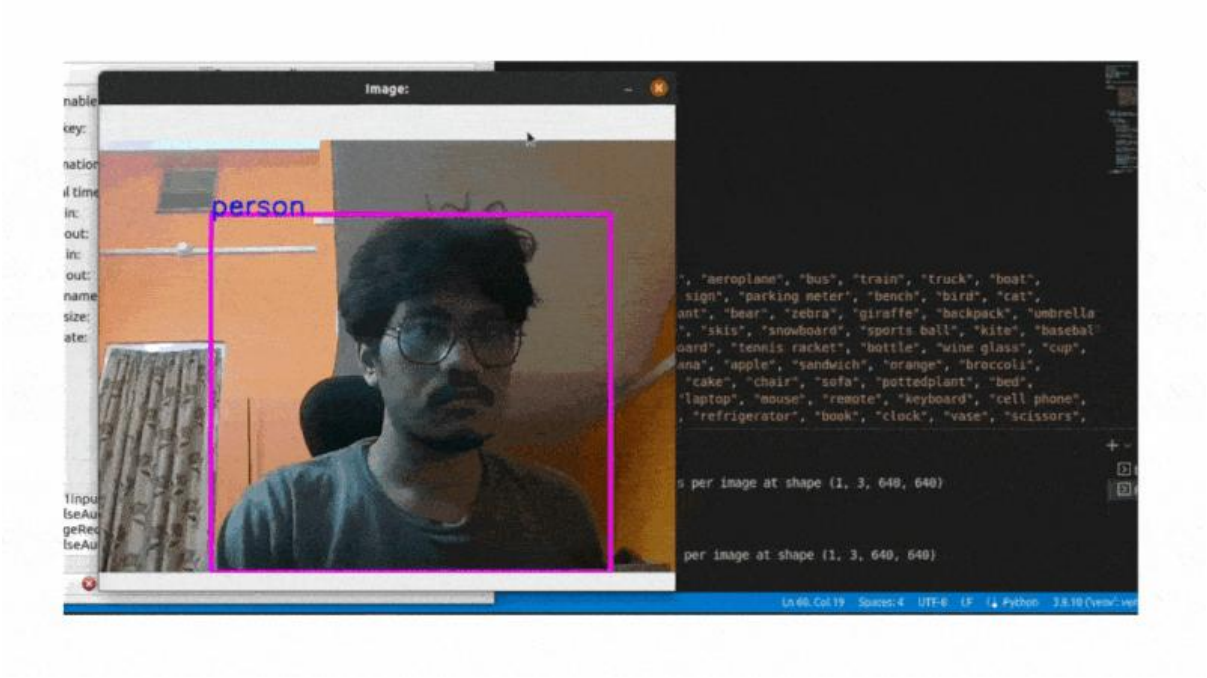
Frame ni grid ga divide chesi
Each grid object detect chestundi
Bounding box + confidence score generate chestundi

Cyber Forensics lo use cases:

- Weapon detection
- Vehicle detection
- Person tracking
- Suspicious object identification

YOLO Working Principle





Requirements:

- Python 3.x
- OpenCV
- Pre-trained YOLO model (YOLOv3 / YOLOv5)
- Webcam

Install libraries:
pip install opencv-python numpy

Procedure

Step1 : Download YOLO Files
Download:

- yolov3.weights
- yolov3.cfg
- coco.names

(Pre-trained model for object detection)

Step2 : Load YOLO Model

```
import cv2
import numpy as np

net = cv2.dnn.readNet("yolov3.weights", "yolov3.cfg")

with open("coco.names", "r") as f:
    classes = [line.strip() for line in f.readlines()]
```

```
layer_names = net.getLayerNames()
output_layers = [layer_names[i - 1] for i in net.getUnconnectedOutLayers()]
```

Step3 : Start Live Video Detection

```
cap = cv2.VideoCapture(0)

while True:
    ret, frame = cap.read()
    height, width, channels = frame.shape

    blob = cv2.dnn.blobFromImage(frame, 0.00392, (416, 416),
                                 (0, 0, 0), True, crop=False)
    net.setInput(blob)
    outs = net.forward(output_layers)

    for out in outs:
        for detection in out:
            scores = detection[5:]
            class_id = np.argmax(scores)
            confidence = scores[class_id]

            if confidence > 0.5:
                center_x = int(detection[0] * width)
                center_y = int(detection[1] * height)
                w = int(detection[2] * width)
                h = int(detection[3] * height)

                x = int(center_x - w / 2)
                y = int(center_y - h / 2)

                cv2.rectangle(frame, (x, y), (x + w, y + h),
                              (0, 255, 0), 2)
                cv2.putText(frame, classes[class_id],
                            (x, y - 10),
                            cv2.FONT_HERSHEY_SIMPLEX,
                            0.5, (0, 255, 0), 2)

    cv2.imshow("YOLO Object Detection", frame)

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

```
cap.release()
cv2.destroyAllWindows()
```

Observations:

- Multiple objects detected simultaneously
- Class labels displayed (person, car, chair, etc.)
- Bounding boxes drawn
- Real-time detection achieved

Result:

AI-based object detection successfully implemented using YOLO.
Live video analytics performed with high accuracy and speed.

Forensic Applications:

- CCTV crime scene analysis
- Suspect identification
- Weapon detection
- Crowd analysis
- Automated surveillance systems

Limitations:

- Requires high system performance
- Lighting conditions affect detection
- Pre-trained model limited to COCO dataset classes

Viva Questions:

1. What is YOLO?
2. Difference between YOLO and Haar Cascade?
3. What is Confidence Score?
4. What is Non-Max Suppression?
5. Why YOLO suitable for real-time detection?