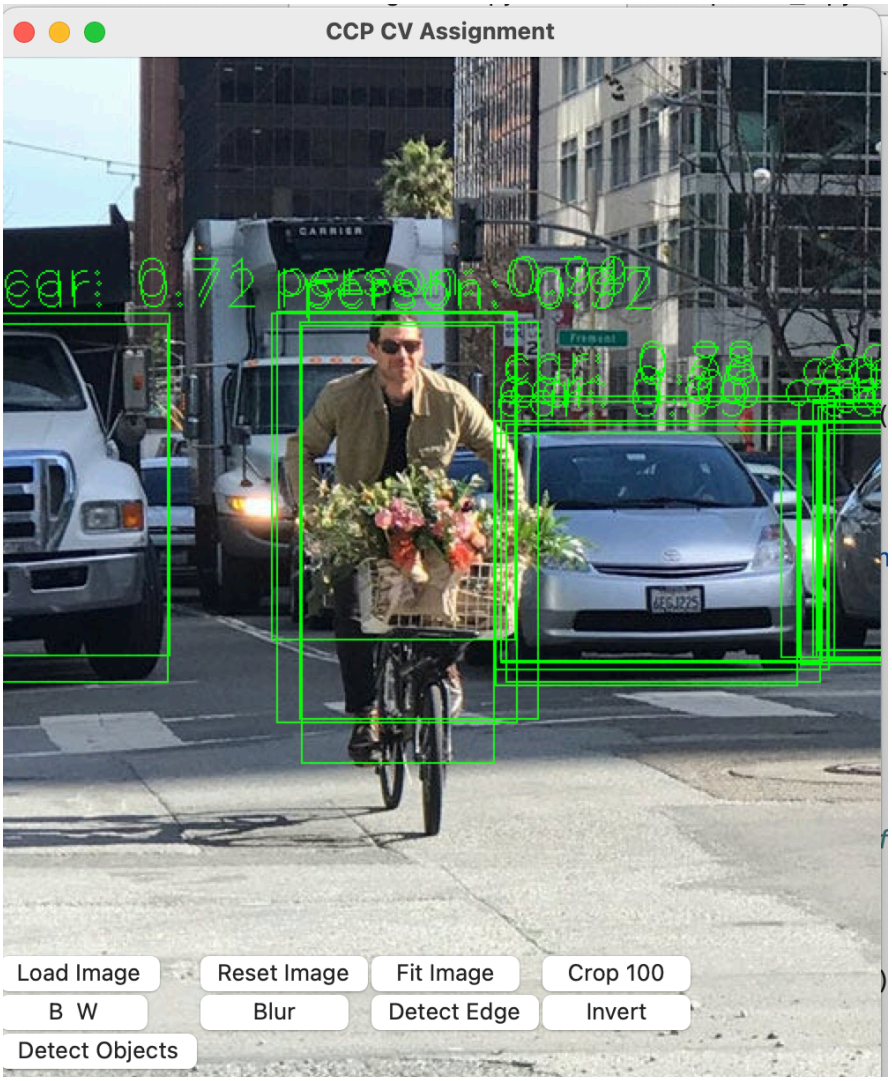
Activity 3 & 4

```python
def OnDetect(self, event):
    # Prepare input image for YOLO model
    img = self.img_raw

    height, width = img.shape[:2]

    # img = img.astype(np.float32)

    blob = cv2.dnn.blobFromImage(img, 1/255, (416, 416), swapRB=True, crop=False)
    self.net.setInput(blob)

    # Get detection results
    outs = self.net.forward(self.net.getUnconnectedOutLayersNames())

    # Process detection results
    for out in outs:
        for detection in out:
            scores = detection[5:]
            class_id = scores.argmax()
            confidence = scores[class_id]
            if confidence > 0.7:  # Set a confidence threshold
                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)

                # Draw bounding box and class label
                cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 1)
                label = f'{self.classes[class_id]}: {confidence:.2f}'
                cv2.putText(img, label, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 1)

    self.ShowCV2Image(img)
```



[1]:

```python
import cv2
import numpy as np

# Capture video feed
cap = cv2.VideoCapture(0)

# Read the frame
ret1, frame1 = cap.read()
prev_gray = cv2.cvtColor(frame1, cv2.COLOR_BGR2GRAY)

while cap.isOpened():
    ret, frame2 = cap.read()
    current_gray = cv2.cvtColor(frame2, cv2.COLOR_BGR2GRAY)

    # Calculate optical flow
    flow = cv2.calcOpticalFlowFarneback(prev_gray, current_gray, None, 0.5, 3, 15, 3, 5, 1.2, 0)

    # Draw motion vectors
    motion_image = np.copy(frame2)
    for y in range(0, motion_image.shape[0], 10):
        for x in range(0, motion_image.shape[1], 10):
            dx, dy = flow[y, x]
            cv2.arrowedLine(motion_image, (x, y), (int(x+dx), int(y+dy)), (0, 0, 255), 1)

    prev_gray = current_gray

    # cv2.imshow('Video', frame1)
    # cv2.imshow('Gray Video', prev_gray)
    cv2.imshow('Optical Flow', motion_image)

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

cap.release()
cv2.destroyAllWindows()
cv2.waitKey(1)
```