```
            firstFrame = gray
            continue

        # compute the absolute difference between the current frame and first
frame
        frameDelta = cv2.absdiff(firstFrame, gray)
        thresh = cv2.threshold(frameDelta, 25, 255, cv2.THRESH_BINARY)[1]

        # dilate the thresholded image to fill in holes, then find contours on
thresholded image
        thresh = cv2.dilate(thresh, None, iterations=2)
        # cnts = cv2.findContours(thresh.copy(),
cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
        # cnts = cnts[0] if imutils.is_cv2() else cnts[1]
        contours, heirarchy = cv2.findContours(thresh.copy(),
cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
        cnts = max(contours, key=cv2.contourArea)


        # loop over the contours identified
        contourcount = 0
        for c in cnts:
            contourcount =   contourcount + 1

           # if the contour is too small, ignore it
            if cv2.contourArea(c) < args["min_area"]:
                continue

            # compute the bounding box for the contour, draw it on the frame,
            (x, y, w, h) = cv2.boundingRect(c)
            initBB2 =(x,y,w,h)


            prott1 = r'MobileNetSSD_deploy.prototxt'
            prott2 = r'mobilenet_iter_73000.caffemodel'
            net = cv2.dnn.readNetFromCaffe(prott1, prott2)

            CLASSES = ["person"]
            COLORS = np.random.uniform(0, 255, size=(len(CLASSES), 3))

            trackbox = frame[y:y+h, x:x+w]
            trackbox = cv2.resize(trackbox, (224, 224))
            cv2.imshow('image',trackbox)
            blob = cv2.dnn.blobFromImage(cv2.resize(trackbox, (300,
300)),0.007843, (300, 300), 127.5)
```