A PROJECT REPORT
On
REAL-TIME OBJECT DETECTION AND TRACKING USING DEEP LEARNING AND OPENCV

submitted in partial fulfillment for the degree

Bachelor of Technology
in
Information Technology

Dr. AMBEDKAR INSTITUTE OF TECHNOLOGY FOR HANDICAPPED, U.P. , KANPUR

Under the guidance of

Mr. ABHISHEK PRABHKAR SIR
Mr. KAPIL PANDEY SIR

PREPARED BY

Hariom Nathani(2001660130025)

DECLARATION

We hereby declare that the project submitted by us is our own work and that to the best of our knowledge. It contains no material previously published or written by another person, nor material which to a substantial extent has been expected for the award of our other degree or diploma of the university or other institute of higher learning except where due acknowledgement has been made in the text.

Signature:
Name: Hariom Nathani
Roll no: 2001660130025

ACKNOWLEDGEMENT

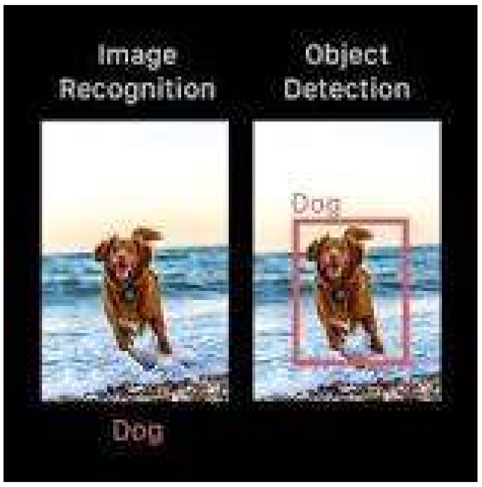
We express our deep sense of gratitude to our project guide **Mr. Abhishek Prabhakar Sir and Mr. Kapil Pandey Sir** for their valuable guidance. We would like to express our feelings of thankfulness to our project guide for giving his constant encouragement that we required during our project. We are thankful to him for his gentle encouragement and pain taken in while guiding us through the project duration.We are also thankful to all the faculty members of the IT Department for their encouragement and support during our project duration.

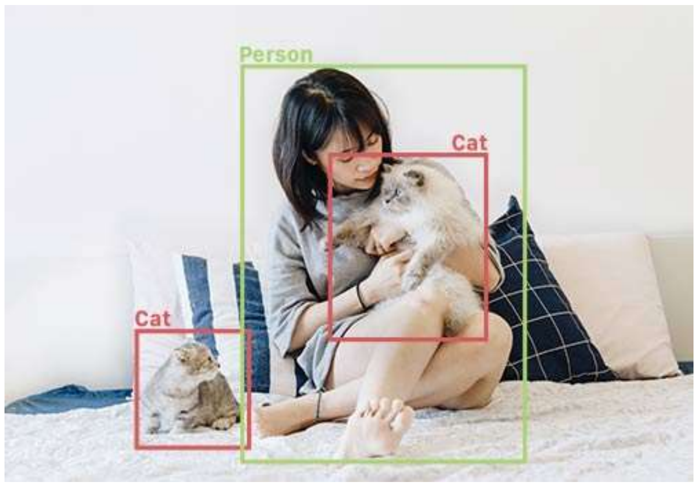
Hariom Nathani (2001660130025)

TABLE OF CONTENT

INTRODUCTION

- Object detection is a computer vision technique that allows us to identify objects in an image or video. This kind of identification and object detection can be used to count objects in a scene and determine and track their precise locations, all while accurately labeling them.
- Object detection is commonly confused with image recognition, so before we processed, it's important that we clarify the distinction between them.
- Image recognition assigns a label to an image. A picture of a dog receives the label "dog". A picture of two dogs, still receives the label "dog". Object detection, on the other hand, draws a box around each dog and labels the box "dog". The model predicts where each object is and what label should be applied. In that way, object detection provides more information about an image than recognition.





Abstract

Deep learning has gained a tremendous influence on how the world is adapting to Artificial Intelligence since past few years. Some of the popular object detection algorithms are Region-based Convolutional Neural Networks (RCNN), Faster-RCNN, Single Shot Detector (SSD) and You Only Look Once (YOLO). Amongst these, Faster-RCNN and SSD have better accuracy, while YOLO performs better when speed is given preference over accuracy. Deep learning combines SSD and Mobile Nets to perform efficient implementation of detection and tracking. This algorithm performs efficient object detection while not compromising on the performance.

PURPOSE

The mainpurpose of object detection is to scan digital images or real-life scenarios of every object, separate them, and analyze their necessary features for real-time predictions. Object detection is a part of the overall data architecture of a company or organization.

DETAILS ABOUT ALL THE FUNCTIONS USED IN THE PROJECT

1. **OpenCV**:-OpenCV is the huge open-source library for computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify an image pattern and its various features we use vector space and perform mathematical operations on these features.

2. **YOLO ALGORITHM:-**The YOLO algorithm works by dividing the image grids, each having an equal dimensional region of SxS. Each of these N grids is responsible for the detection and localization of the object it contains

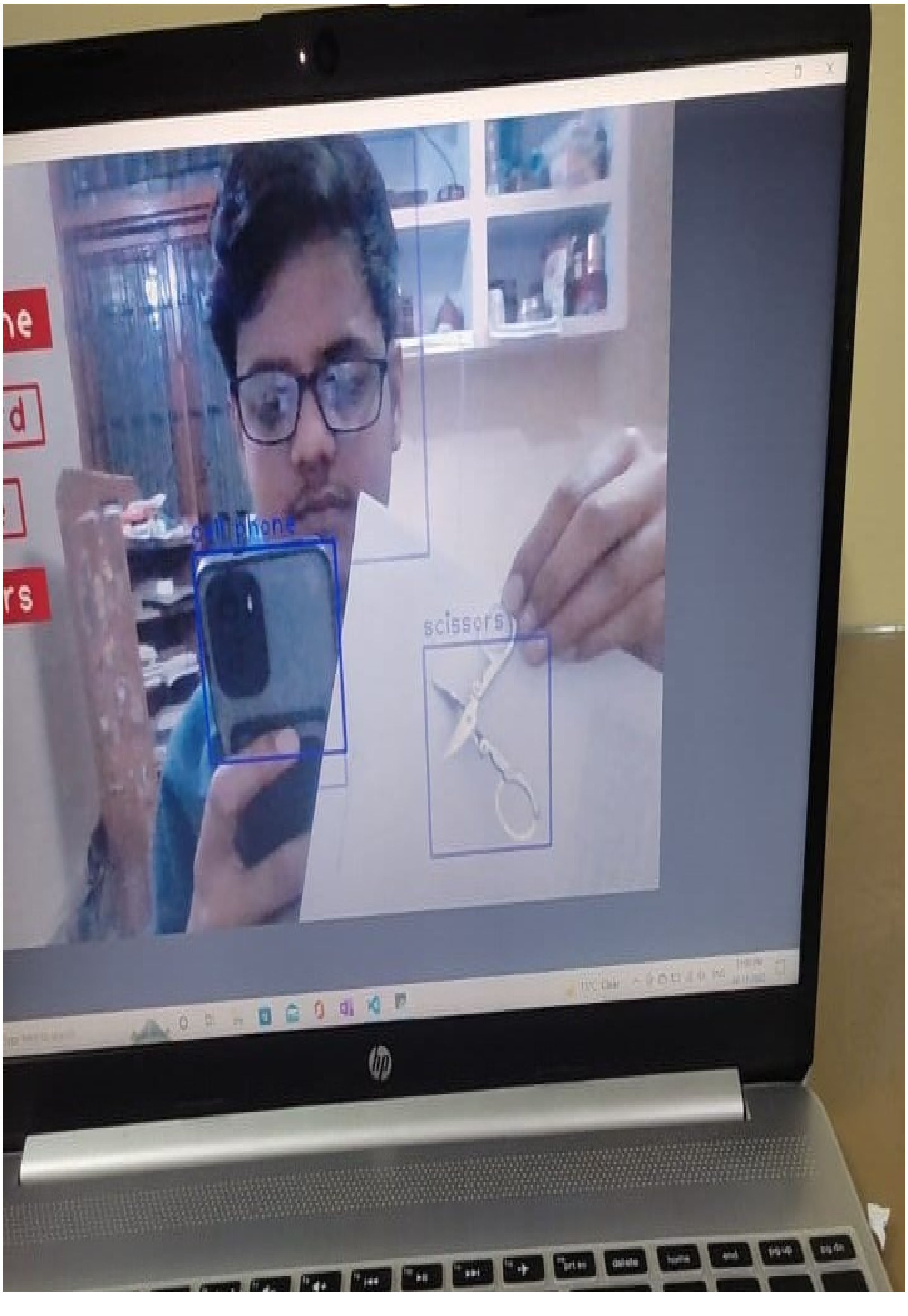
3. **OpenCV DNN Module:-**We all know OpenCV as one of the bestcomputer vision libraries out there. Additionally, it also has functionalities for running deep learning inference as well. The best part is supporting the loading of different models from different frameworks using which we can carry out several deep learning functionalities.

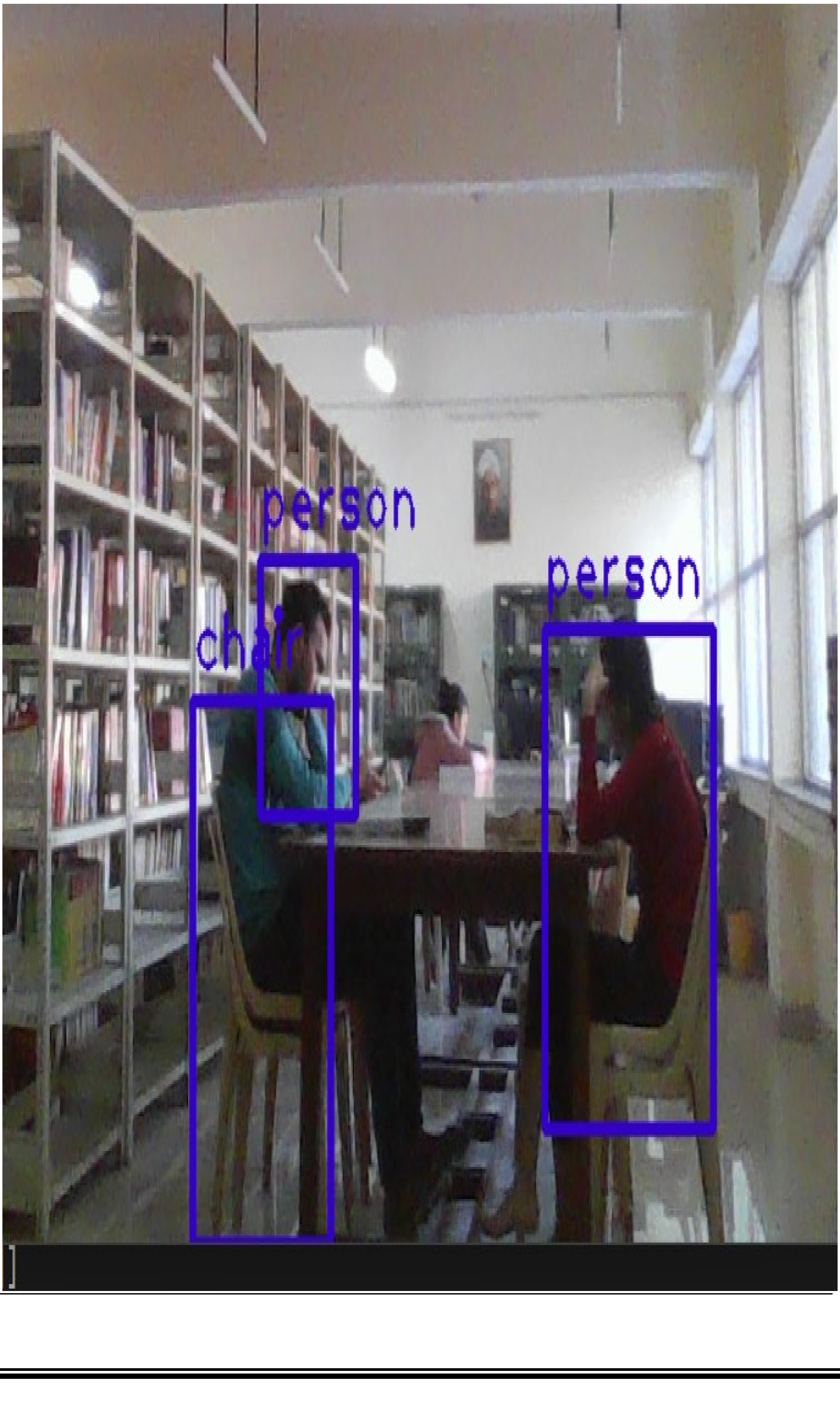
CODE

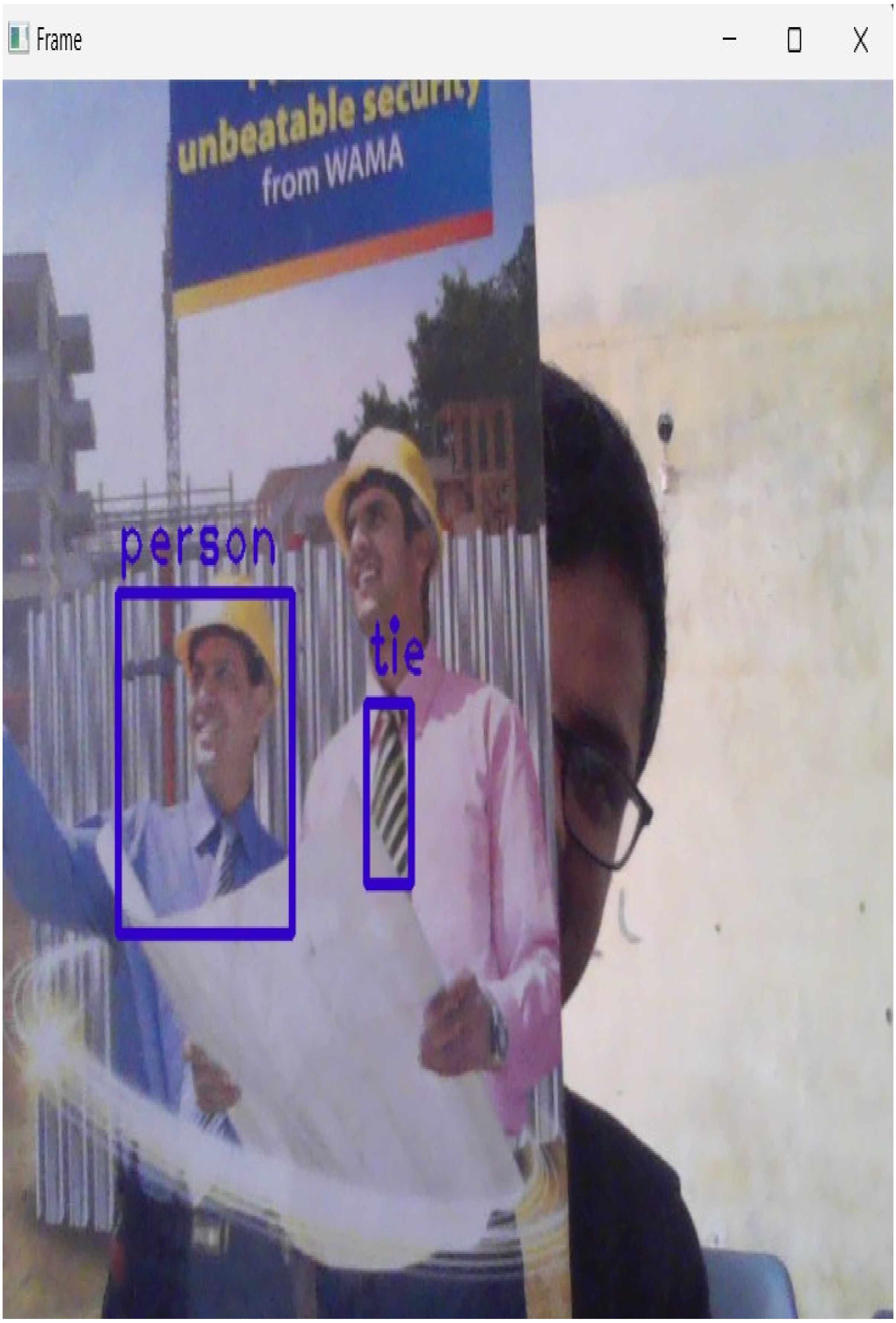
```
import cv2
# import numpy as np
From gui_buttonsimport Buttons

# INITIALIZING BUTTONS
button = Buttons()
button.add_button("person", 20, 20)
button.add_button("cell phone", 20, 100)
button.add_button("keyboard", 20, 180)
button.add_button("remote", 20, 260)
button.add_button("scissors", 20, 340)

colors= button.colors
#Opencv DNN
net=cv2.dnn.readNet("dnn_model/yolov4-tiny.weights","dnn_model/yolov4-
tiny.cfg")
model=cv2.dnn_DetectionModel(net)
model.setInputParams(size=(320,320),scale=1/255)

# load class lists
classes = []
with open("dnn_model/classes.txt", "r") as file_object:
      for class_name in file_object.readlines():
            class_name = class_name.strip()
            classes.append(class_name)
print("objects list")
print(classes)

#initialize camera
cap=cv2.VideoCapture(0)
# increase resolution of camera ..change buffer size ,exposure of camera
cap.set(cv2.CAP_PROP_FRAME_WIDTH, 1280)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 720)
# FULL HD 1920 x 1080

# button_person = False

Def click_button(event, x, y, flags, params):
      Global button_person
      if event == cv2.EVENT_LBUTTONDOWN:
            # print(x, y)
            button.button_click(x, y)
            '''
            polygon =np.array([[(20, 20), (220, 20), (220, 70), (20, 70)]])

            is_inside = cv2.pointPolygonTest(polygon, (x, y), False)
            if is_inside>0:
                  print("We are clicking inside the button", x, y)

                  if button_person is False:
                        button_person = True
                  else:
                        button_person = False

                  print("Now Button person is:", button_person)
            '''

# create window
cv2.namedWindow("Frame")
cv2.setMouseCallback("Frame", click_button)

while True:
  # get frame
  ret,frame=cap.read()
  #get active button list
  active_buttons= button.active_buttons_list()
  print("Active buttons", active_buttons)


#   objectdectection
  (class_ids, scores, bboxes) = model.detect(frame, confThreshold=0.3, 
nmsThreshold= .4)
  For class_id, score, bboxinzip(class_ids, scores, bboxes):
      (x, y, w, h)=bbox
      class_name = classes[class_id]
      color= colors[class_id]

      if class_nameinactive_buttons:
            cv2.putText(frame, class_name, (x, y - 10), cv2.FONT_HERSHEY_PLAIN, 
2, (200, 0, 50), 2)
            cv2.rectangle(frame, (x, y), (x + w, y + h), (200, 0, 50), 3)

      #print(x, y, w ,h)

      # print(class_name)

      '''if class_name == "person":
            cv2.putText(frame, class_name, (x, y - 10), cv2.FONT_HERSHEY_PLAIN, 
2, (200, 0, 50), 2)
            cv2.rectangle(frame, (x, y), (x + w, y + h), (200, 0, 50), 3)
'''
```

```
      '''
      if class_name == "cell phone":
            cv2.putText(frame, class_name, (x, y - 10), cv2.FONT_HERSHEY_PLAIN,
2, (200, 0, 50), 2)
            cv2.rectangle(frame, (x, y), (x + w, y + h), (200, 0, 50), 3)
'''
      '''
      if class_name == "person" and button_person is True:
            cv2.putText(frame, class_name, (x, y - 10), cv2.FONT_HERSHEY_PLAIN,
2, (200, 0, 50), 2)
            cv2.rectangle(frame, (x, y), (x + w, y + h), (200, 0, 50), 3)
'''



      '''
    # CREATE BUTTON
#   cv2.rectangle(frame, (20, 20),(220, 70),(0, 0, 200), -1)
  polygon =np.array([[(20, 20), (220, 20), (220, 70), (20, 70)]])
  cv2.fillPoly(frame, polygon, (0, 0, 200))
  cv2.putText(frame, "Person",(30, 60),cv2.FONT_HERSHEY_PLAIN, 3, (255, 255,
255), 3)
'''




#   print("class ids",, class_ids)
#   print("scores", scores)
#   print("bboxes", bboxes)

  #DISPLAY BUTTON
  button.display_buttons(frame)


  cv2.imshow("Frame",frame)

  key = cv2.waitKey(1)
  if key==27:
      break;

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

SNAPSHOT OF OUTPUT RESULT




person
person
chair


Frame
unbeatable security
from WAMA
person
tie

CONCLUSION

Object detection is a key ability for most computer and robot vision system. Although great progress has been observed in the last years, and some existing techniques are now part of many consumer electronics (e.g., face detection for auto-focus in smartphones) or have been integrated in assistant driving technologies, we are still far from achieving human-level performance, in particular in terms of open-world learning. It should be noted that object detection has not been used much in many areas where it could be of great help. As mobile robots, and in general autonomous machines, are starting to be more widely deployed (e.g., quad-copters, drones and soon service robots), the need of object detection systems is gaining more importance. Finally, we need to consider that we will need object detection systems for nano-robots or for robots that will explore areas that have not been seen by humans, such as depth parts of the sea or other planets, and the detection systems will have to learn to new object classes as they are encountered. In such cases, a real-time open-world learning ability will be critical

REFERENCES

1. https://pysource.com/2022/01/10/build-your-object-detection-software-crash-course/
2. https://www.geeksforgeeks.org
3. https://github.com/