**Programme: Higher Diploma in Artificial Intelligence and Robotics**

**Programme code: EG114728**

|  |  |  |
| --- | --- | --- |
| **Official Use** | | |
|  | Full Mark | Mark |
| **Total** | 100% |  |

**Module: AI and Programming**

**Module Code: MBS 3523**

**Assessment: Assignment 2**

**Due Date: 17 April 2023**

|  |  |  |
| --- | --- | --- |
| **Name:**  **Cheung Tsz Chun Noddy** | **Class:**  **EG114728** | **Student no.:**  **220171174** |

**NOTES:**

* **Answer all questions.**
* **Full mark of this paper is 100.**
* **Attach your programs with this paper.**

**Submission deadline: 17 April 2023 5:00 pm**

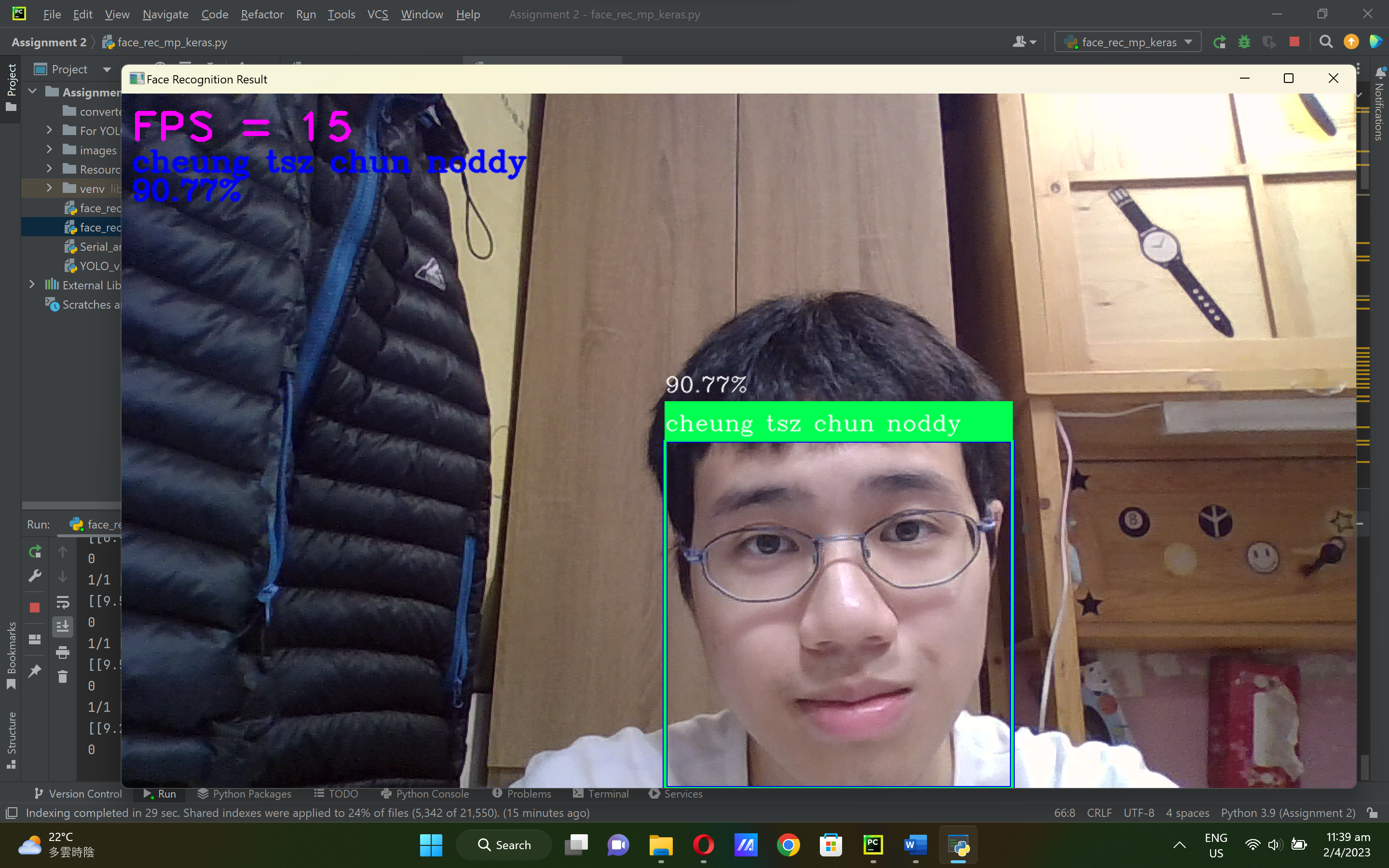
“*I declare that this assessment is my own work and was not copied from any other person”*

*Signed: \_\_\_\_\_\_\_\_Cheung Tsz Chun\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_5/4\_\_\_\_\_\_\_*

**NOTE:**

* For this assignment (Question 1 and 2), you are required to demonstrate the operation of your programs with specific tasks/requirements that will be given during the demonstration section!
* You may bring your own notebook computer or use the computer in lab (webcam will be provided).
* You have 3 minutes for each question demonstration. That means a total of 6 minutes for the whole assignment. No extra time will be given unless there is a technical problem. Be aware and well prepare for the demonstration.
* The demonstration period is on **17 April 2023, 09:30 – 11:30**.
* If you cannot demonstrate any code, a reassessment will be arranged on 24 April 2023, 09:30 – 11:30. The maximum marks from the reassessment will be 40!
* If face-to-face lesson cannot be arranged within reasonable timeframe, you will be required to make a video clip to demonstrate the requirements.

|  |
| --- |
| **Question 1 (50%)** |
| With reference to ***face\_rec\_autocollection.py and face\_rec\_mp\_keras.py***, write a python code with below requirements:   1. Demonstrate your code with your own face with your **name and score** on top left corner of the bounding box; 2. your code should at least recognize THREE persons. The known person’s **name and score** should be displayed on top left corner of the bounding box; 3. If there is any person that is not recognizable, “Unknown Person” should be displayed on the top left corner of the bounding box; 4. Screen capture a screen with your own face shown.   *# Other Requirements:*  *# Save the code on your Github as* ***MBS3523-Asn2-Q1.py***  *# Print the code and attach to this paper*  *# Screen capture a screen with your own face shown.*  <https://github.com/noddycheung/cv2/blob/e3c184d12800fce32f1f713ee41d5f350b393c0e/MBS3523-Asn2-Q1.py>  import tensorflow as tf  from tensorflow import keras  import cv2  import mediapipe as mp  from keras.models import load\_model  import numpy as np  import time  detectFace = mp.solutions.face\_detection.FaceDetection(min\_detection\_confidence=0.6)  cam = cv2.VideoCapture(0)  cam.set(3, 1280)  cam.set(4, 720)  font = cv2.FONT\_HERSHEY\_COMPLEX  model = load\_model('Resources/keras\_model.h5')  # variables for FPS  t\_old = 0  t\_new = 0  # Define known persons and their labels  known\_persons = {  'cheung tsz chun noddy': 0,  'oscar': 1,  'kai ming': 2,  'Unknown Person' :3  }  def get\_className(classNo):  for name, label in known\_persons.items():  if label == classNo:  return name  return 'Unknown Person'  while True:  ret, img = cam.read()  imgRGB = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)  results = detectFace.process(imgRGB)  # print(results.detections)  if results.detections != None:  for face in results.detections:  # drawFace.draw\_detection(img, face)  boundingBox = face.location\_data.relative\_bounding\_box  x1 = int(boundingBox.xmin \* 1280)  y1 = int(boundingBox.ymin \* 720)  x2 = int((boundingBox.xmin + boundingBox.width) \* 1280)  y2 = int((boundingBox.ymin + boundingBox.height) \* 720)  pt1 = (x1, y1)  pt2 = (x2, y2)  cv2.rectangle(img, pt1, pt2, (255, 0, 0), 3)  crop\_img = img[y1:y2, x1:x2]  imgResize = cv2.resize(crop\_img, (224, 224))  imgReshape = imgResize.reshape(1, 224, 224, 3)  prediction = model.predict(imgReshape)  print(prediction)  classIndex = np.argmax(prediction)  print(classIndex)  probabilityValue = np.amax(prediction)  # print(probabilityValue)  if classIndex == 0 or 1 or 2:  cv2.rectangle(img, (x1, y1), (x2, y2), (80, 255, 0), 2)  cv2.rectangle(img, (x1, y1 - 40), (x2, y1), (80, 255, 0), -2)  cv2.putText(img, str(get\_className(classIndex)), (x1, y1 - 10), font, 0.75, (255, 255, 255), 1, cv2.LINE\_AA)  cv2.putText(img, str(round(probabilityValue \* 100, 2)) + "%", (10, 110), font, 1, (255, 0, 0), 2,  cv2.LINE\_AA)  cv2.putText(img, str(round(probabilityValue \* 100, 2)) + "%", (x1, y1 - 50), font, 0.75, (255, 255, 255), 1,  cv2.LINE\_AA)  cv2.putText(img, str(get\_className(classIndex)), (10, 80), font, 1, (255, 0, 0), 2,  cv2.LINE\_AA)  else:  cv2.putText(img, str(get\_className(classIndex)), (10, 110), font, 1.5, (255, 0, 0), 2, cv2.LINE\_AA)  cv2.putText(img, str(get\_className(classIndex)) , (x1, y1 - 10), font, 0.75, (255, 255, 255), 1,  cv2.LINE\_AA)  if results.detections == None:  cv2.putText(img, str('Unknown Person'), (10, 110), font, 1.5, (255, 0, 0), 2, cv2.LINE\_AA)  # Calculate FPS and display on upper left  t\_new = time.time()  fps = 1 / (t\_new - t\_old)  t\_old = t\_new  cv2.putText(img, 'FPS = ' + str(int(fps)), (10, 50), cv2.FONT\_HERSHEY\_PLAIN, 3,  (255, 0, 255), 3)  cv2.imshow("Face Recognition Result", img)  if cv2.waitKey(1) & 0xff == 27:  break  cam.release()  cv2.destroyAllWindows() |



|  |
| --- |
| **Question 2 (50%)** |
| With reference to ***Serial\_arduino\_servotrack.py*** and ***YOLO\_v3.py*** write a python code and demonstrate:   1. the tracking result, with correct name shown, when ONE randomly selected object from the 80 COCO object list is specified.   Python code: <https://github.com/noddycheung/cv2/blob/b8b2bf87d78d5e41e67b6b645ffd4c448da8fd71/MBS3523-Asn2-Q2.py>  import cv2  import serial,time  import numpy as np  classesFile = 'For YOLO/For YOLO/coco80.names'  classes = open(classesFile, 'r').read().splitlines()  confThreshold = 0.6  net = cv2.dnn.readNetFromDarknet('For YOLO/For YOLO/yolov3-320.cfg', 'For YOLO/For YOLO/yolov3-320.weights')  net.setPreferableBackend(cv2.dnn.DNN\_BACKEND\_OPENCV)  net.setPreferableTarget(cv2.dnn.DNN\_TARGET\_CPU)  class\_of\_interest = 67 # index of "remote"  cap=cv2.VideoCapture(0, cv2.CAP\_DSHOW)  #fourcc= cv2.VideoWriter\_fourcc(\*'XVID')  ArduinoSerial=serial.Serial('com6',9600,timeout=0.1)  #out= cv2.VideoWriter('face detection4.avi',fourcc,20.0,(640,480))  time.sleep(1)  while cap.isOpened():  ret, frame= cap.read()  height, width, ch = frame.shape  blob = cv2.dnn.blobFromImage(frame, 1 / 255, (320, 320), (0, 0, 0), swapRB=True, crop=False)  net.setInput(blob)  outputs = net.forward(net.getUnconnectedOutLayersNames())  boxes, confidences = [], []  for output in outputs:  for detection in output:  scores = detection[5:]  class\_id = np.argmax(scores)  confidence = scores[class\_id]  if confidence > confThreshold and class\_id == class\_of\_interest:  center\_x, center\_y, w, h = map(int, detection[:4] \* [width, height, width, height])  x, y = center\_x - w // 2, center\_y - h // 2  boxes.append([x, y, w, h])  confidences.append(confidence)  indexes = cv2.dnn.NMSBoxes(boxes, confidences, confThreshold, 0.4)  font, colors = cv2.FONT\_HERSHEY\_PLAIN, np.random.uniform(0, 255, size=(len(boxes), 3))  if len(indexes) > 0:  for i in indexes.flatten():  x, y, w, h = boxes[i]  label, color = str(classes[class\_of\_interest]), colors[i]  cv2.rectangle(frame, (x, y), (x + w, y + h), color, 2)  cv2.putText(frame, f"{label}", (x, y + 20), font, 2, (255, 255, 255), 2)  for x,y,w,h in boxes:  #sending coordinates to Arduino  string='X{0:d}Y{1:d}'.format((x+w//2),(y+h//2))  print(string)  ArduinoSerial.write(string.encode('utf-8'))  #plot the center of the face  cv2.circle(frame,(x+w//2,y+h//2),2,(0,255,0),2)  #plot the roi  cv2.rectangle(frame,(x,y),(x+w,y+h),(0,0,255),3)  #plot the squared region in the center of the screen  cv2.rectangle(frame,(640//2-30,480//2-30),  (640//2+30,480//2+30),  (255,255,255),3)  #out.write(frame)  cv2.imshow('img',frame)  #cv2.imwrite('output\_img.jpg',frame)  '''for testing purpose  read= str(ArduinoSerial.readline(ArduinoSerial.inWaiting()))  time.sleep(0.05)  print('data from arduino:'+read)  '''  # press q to Quit  if cv2.waitKey(10)&0xFF== ord('q'):  break  cap.release()  cv2.destroyAllWindows()  Arduino code:  <https://github.com/noddycheung/cv2/blob/304a16f9eb6c5c98ab8ed1bcee25145d5cadde02/MBS3523-Asn2-Q2.arduino>  #include<Servo.h>  Servo x, y;  int width = 640, height = 480; // total resolution of the video  int xpos = 90, ypos = 90; // initial positions of both Servos  void setup() {  Serial.begin(9600);  x.attach(9);  y.attach(10);  // Serial.print(width);  //Serial.print("\t");  //Serial.println(height);  x.write(xpos);  y.write(ypos);  }  const int angle = 2; // degree of increment or decrement  void loop() {  if (Serial.available() > 0)  {  int x\_mid, y\_mid;  if (Serial.read() == 'X')  {  x\_mid = Serial.parseInt(); // read center x-coordinate  if (Serial.read() == 'Y')  y\_mid = Serial.parseInt(); // read center y-coordinate  }  /\* adjust the servo within the squared region if the coordinates  is outside it  \*/  if (x\_mid > width / 2 + 30)  xpos -= angle;  if (x\_mid < width / 2 - 30)  xpos += angle;  if (y\_mid < height / 2 + 30)  ypos += angle;  if (y\_mid > height / 2 - 30)  ypos -= angle;  // if the servo degree is outside its range  if (xpos >= 180)  xpos = 180;  else if (xpos <= 0)  xpos = 0;  if (ypos >= 180)  ypos = 180;  else if (ypos <= 0)  ypos = 0;  x.write(xpos);  y.write(ypos);  // used for testing  //Serial.print("\t");  // Serial.print(x\_mid);  // Serial.print("\t");  // Serial.println(y\_mid);  }  }    Python code: <https://github.com/noddycheung/cv2/blob/b8b2bf87d78d5e41e67b6b645ffd4c448da8fd71/MBS3523-Asn2-Q2.py>  Arduino code:  <https://github.com/noddycheung/cv2/blob/304a16f9eb6c5c98ab8ed1bcee25145d5cadde02/MBS3523-Asn2-Q2.arduino>  NOTE:  You need to connect your Arduino Uno board with two servos and a webcam to the PC/notebook computer. After tuning your code, the servo/webcam combo should track the selected object given when it is within the FOV (field of view).  *# Other Requirements:*  *# Save the code on your Github as* ***MBS3523-Asn2-Q2.py***  *# Print the code and attach to this paper*  *# Take a picture with the object, the code running and the object is being tracked.* |

**~ End of Question ~**