Hand Gesture Detection

التعرف على إيماءات اليد

**عمل الطلاب**

محمود عماد الحجو

ياسر عمار كنج

وائل ياسين زعيتر

**مقدمة:**

التقنيات والمكاتب المستخدمة:

* **لغة البرمجة (Python):**

هي لغة برمجة، عالية المستوى، تتميز ببساطة كتابتها وقراءتها، سهلة التعلم، تستخدم أسلوب البرمجة الكائنية، وهي مفتوحة المصدر، وقابلة للامتداد. تعتبر لغة بايثون لغة مفسرة، متعددة الاستخدامات وتستخدم بشكل واسع في العديد من المجالات، كبناء البرامج المستقلة باستخدام الواجهات الرسومية المعروفة وفي عمل تطبيقات الويب، بالإضافة إلى استخدامها كلغة برمجة نصية للتحكم في أداء بعض من أشهر البرامج المعروفة أو في بناء برامج ملحقة لها. وبشكل عام يمكن استخدام بايثون لبرمجة البرامج البسيطة للمبتدئين، ولإنجاز المشاريع الضخمة كأي لغة برمجية أخرى في نفس الوقت. غالباً ما يُنصح المبتدؤون في ميدان البرمجة بتعلم هذه اللغة لأنها من بين أسرع اللغات البرمجية تعلماً.

* **مكتبة (OpenCV):**

هي المكتبة البرمجية المفتوحة للرؤية الحاسوبية هي مكتبة اقترانات برمجية تهدف بشكل أساسي لتطوير الرؤية الحاسوبية، طورتها شركة أنتل (Intel) وتعاقب على دعمها بعد ذلك شركتي ويلو غارج (Willow Garge) للروبوتيات و إتسيز (Itseez) وأخير شركة إنتل مجددا بعد استيلائها على شركة Itseez منذ 2016. المكتبة مجانية كونها تحت رخصة المصدر المفتوح (open source BSD license). ويمكن استخدامها على معظم الأنظمة الحاسوبية التي تدعم لغة (C++) تركز بشكل أساسي على معالجة اللحظية للصور (real-time).

إذا وجدت المكتبة على النظام ما يدعى بدائيون الأداء المتكامل (Integrated Performance Primitives) المطورة من قبل شركة أنتل (Intel) فسوف تستخدم هذه الإجراءات التحسينية ذات الرخصة الخاصة للتسريع من عملها.

* **مكتبة (NumPy):**

هي إضافة على لغة البرمجة بايثون، تٌستخدم للتعامل مع المصفوفات الكبيرة والحقول متعددة المستوى، وكذلك توفر مكتبة كبيرة من الاقترانات الرياضية عالية المستوى للعمل على هذه الحقول والمصفوفات. فكرة نمباي جاءت من الإضافة جيم هوغنن والتي كانت في الأساس مطورة من قِبَل جيم هيوجيونين. وفي عام 2005 قام ترافياس أوليفانت بإنشاء نمباي بميزات ال Numeric وبإضافات واسعة عليها.

* **مكتبة (imutils):**

سلسلة من التوابع الجاهزة تم إنشائها لجعل وظائف معالجة الصور الأساسية مثل الترجمة ، والتدوير ، وتغيير الحجم ، وعرض الصور أسهل مع OpenCV.

**شرح الكود:**

**recognize.py**

**استيراد المكاتب التي نريد استخدامها**

from pyimagesearch.gesture\_recognition import GestureDetector

from pyimagesearch.gesture\_recognition import MotionDetector

import numpy as np

import argparse

import imutils

import cv2

**GestureDetector:**

بايثون كلاس يحتوي التوابع اللازمة لقراءة أصابع اليد والكف في المربع الحيوي الذي يتم تحديده من سطر الأوامر ,وسنقوم لاحقاً بشرحه محتوياته بالتفصيل.

MotionDetector**:**

بايثون كلاس يحتوي التوابع اللازمة لمتابعة حركة أصابع اليد والكف في المربع الحيوي الذي يتم تحديده من سطر الأوامر ,وسنقوم لاحقاً بشرحه محتوياته بالتفصيل.

ap = argparse.ArgumentParser()

ap.add\_argument("-b", "--bounding-box", required=True,

  help="comma separted list of top, right, bottom, left coordinates of hand ROI")

ap.add\_argument("-v", "--video", required=False, help="path to the (optional) video file")

args = vars(ap.parse\_args())

في هذه القطعة نقوم بتعريف أوامر خاصة تظهر في سطر الأوامر عند تشغيل الملف ووضع قواعد محددة لها باستخدام مكتبة (**argparse**) المضمنة في لغة البايثون.

حيث الوسيط الأول الذي يجب إدخاله هو أبعاد المربع الحيوي الذي سنقوم القراءة ضمن حدوده.

والوسيط الثاني هو وسيط اختياري في حال لم نرد القراءة من الكاميرا بل من فيديو محدد موجود على الحاسب فنقوم بوضع مسار الفيديو في هذا الوسيط.

if not args.get("video", False):

  camera = cv2.VideoCapture(0)

# otherwise, grab a reference to the video file

else:

  camera = cv2.VideoCapture(args["video"])

في هذه القطعة نقوم بالتحقق فيما إذا كان المستخدم قد أدخل الوسيط الخاص بالفيديو للقراءة منه, وفي حال لم يقم بإدخاله تتم القراءة من الكاميرا.

(top, right, bot, left) = np.int32(

args["bounding\_box"].split(",")

)

gd = GestureDetector()

md = MotionDetector()

numFrames = 0

gesture = None

values = []

في هذه القطعة نقوم بقراءة أبعاد المربع الحيوي ROI من وتعريف المتغيرات النسخ الصفوف (الكلاسات) التي نحتاجها.

  (grabbed, frame) = camera.read()

  if args.get("video") and not grabbed:

    break

في هذه القطعة نقوم بقراءة الإطار من الكاميرا ونوقف البرنامج في حال عدم وجوده. وإذا وصلنا لنهاية المقطع.

  frame = imutils.resize(frame, width=600)

  frame = cv2.flip(frame, 1)

  clone = frame.copy()

  (frameH, frameW) = frame.shape[:2]

  roi = frame[top:bot, right:left]

  gray = cv2.cvtColor(roi, cv2.COLOR\_BGR2GRAY)

  gray = cv2.GaussianBlur(gray, (7, 7), 0)

في القطعة الأولى نقوم بعكس الفيديو للتخلص من الانعكاس (حالة المرآة).

وفي القطعة الثانية نقوم باستخراج المنقطة الحيوية ROI من إطار الفيديو ثم نقوم بتحويلها لصورة رمادية وتمويهها قليلاً.

  if numFrames < 32:

    md.update(gray)

  else:

    skin = md.detect(gray)

إذا لم نكن قد حصلنا على 32 إطار على الأقل فيستمر الMotionDetector من تحديث الصورة الحالية حتى الوصول لهذه القيمة.

وإذا وصلنا لعدد الأطر الأولي المطلوب فنقوم من التحقق من حركة اليد في الصورة الحالية.

    if skin is not None:

      (thresh, c) = skin

      cv2.drawContours(clone, [c + (right, top)], -1, (0, 255, 0), 2)

      fingers = gd.detect(thresh, c)

ثم نتحقق فيما إذا تم التحقق من وجود اليد وفي حال وجودها نقوم باستخراج العتبة والحدود لهذه اليد (skin) ورسمها على الإطار clone باللون المحدد rgb(0,255,0) وضمن الأبعاد المستخرجة c حيث نقطة البداية هي (right, top).

ثم نقوم باستخدام GestureDetector من قراءة الأصابع وفق المعطيات التي حصلنا عليها (العتبة والحدود).

      if gesture is None:

        gesture = [1, fingers]

      else:

        if gesture[1] == fingers:

          gesture[0] += 1

نعرف متغير يحوي عدد الأصابع التي تم التحقق منها حيث نبد من القيمة 1 كقيمة أولية.  
أما في باقي الحالات عندما نكون قد عرفناه مسبقاً ,  
إذا كان عدد الأصابع نفسه بين الإطار الحالي والسابق فنقوم نزيد عدد الأُطُر بمقدار 1.

          if gesture[0] >= 25:

            if len(values) == 2:

              values = []

            values.append(fingers)

            gesture = None

إذا وصلنا لعدد أطُر مناسب نقوم بملف المصفوفة values التي سنستخدمها لطباعة محتوياتها على المقطع.  
في حال كانت المصفوفة ممتلئة نقوم بإفراغها وملئها بعدد الأصابع المقروءة الحالي. ثم نقوم بمحو قيمة المتغير الذي يحوي عدد هذه الأصابع التي تم تخزينها في المصفوفة values.

  if len(values) > 0:

    GestureDetector.drawBox(clone, 0)

    GestureDetector.drawText(clone, 0, values[0])

    GestureDetector.drawText(clone, 1, "+")

إذا تم قراءة قيمة واحدة لعدد الأصابع على الأقل نقوم بطباعة هذا العدد على المقطع.

  # check to see if there is a second entry in the values list

  if len(values) == 2:

    # draw the second digit, the equal sign, and fianlly the answer

    GestureDetector.drawBox(clone, 2)

    GestureDetector.drawText(clone, 2, values[1])

    GestureDetector.drawText(clone, 3, "=")

    GestureDetector.drawBox(clone, 4, color=(0, 255, 0))

    GestureDetector.drawText(clone, 4, values[0] + values[1], color=(0, 255, 0))

وفي حال وجود عدد ثاني للأصابع تمت قراءته وتخزينه نقوم بجمع العددين وطباعتهما مع الناتج على المقطع.

  cv2.rectangle(clone, (left, top), (right, bot), (0, 0, 255), 2)

  numFrames += 1

رسم حدود المنطقة الحيوية على المقطع وزيادة الإطار الحالي للانتقال للإطار التالي.

  cv2.imshow("Frame", clone)

  key = cv2.waitKey(1) & 0xFF

  if key == ord("q"):

    break

camera.release()

cv2.destroyAllWindows()

إظهار المقطع على الشاشة وتعريف زر خروج "q" لإنهاء البرنامج عند الضغط عليه.

وفي النهاية نقوم بإزالة الكاميرا الحالية ومحو الذاكرة المؤقتة والنسخ المعرفة مسبقاً ثم نقوم بإغلاق جميع النوافذ.