



Program help to edit poses to make a photo like an influencer by artificial intelligence

BADS7203 Image and Video Analytics

Presented by

กชกร เลื่อนสุขสันต์ 6310412001

กอฟฟารี่ ลัดเลีย 6310412019

สุพัตรา ตั้งสกลระหง 6310412028

TABLE OF CONTENTS



01

Introduction

- Goal
- Dataset
- Challenge
- Process in project



02

Literature review

- Human Pose Estimation
- Object Detection



03

Research Methodology

- Data preparation
- Model
 - Blazepose
 - Yolo



04

Result and Show practical application





01 Introduction



Goal :

- Make a program to recommend portraits, which the program will have a function for the photographer to choose the desired pose. and take pictures automatically



Dataset :

- Photo Pose in posture, standing and sitting
Dataset from Printerest



Challenge :

- No has created a program to recommend specific poses for photos yet.

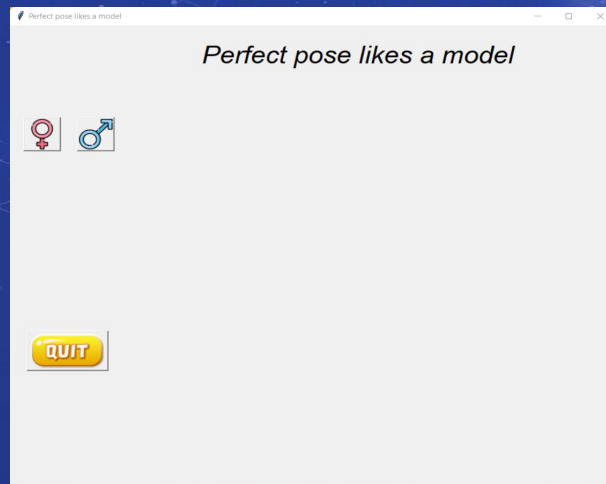
Process Step 1 : Choose gender



male



female



Process

Step 2 : Choose gesture

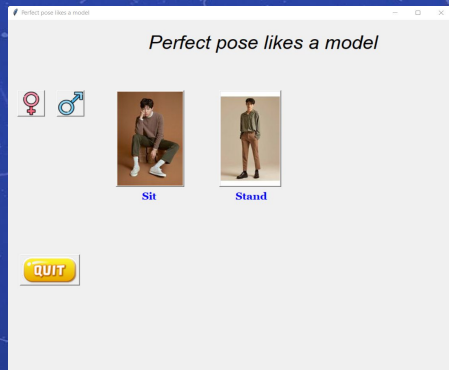


Stand up

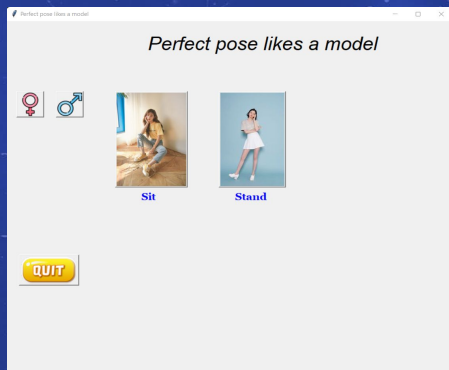


Sit down

Male



Female



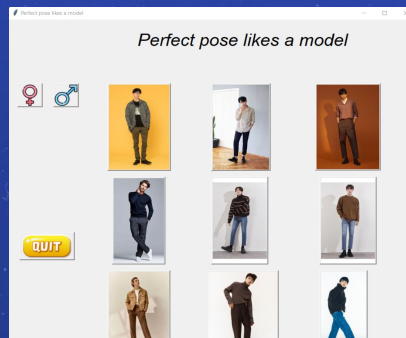
Process

Step 3 : Choose photo

Male



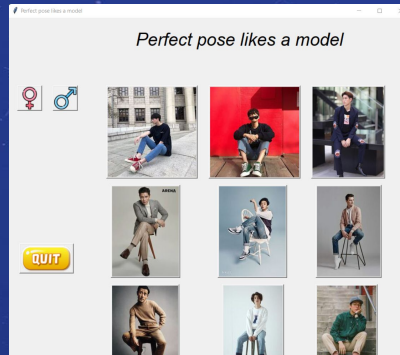
Stand



Male



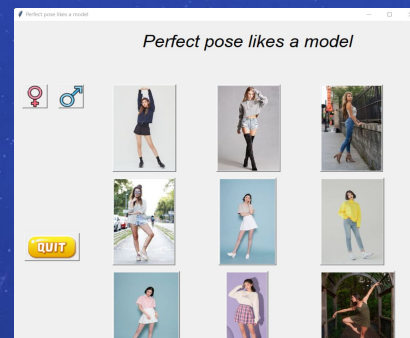
Sit



Female



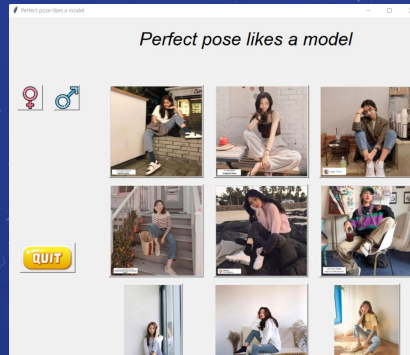
Stand



Female

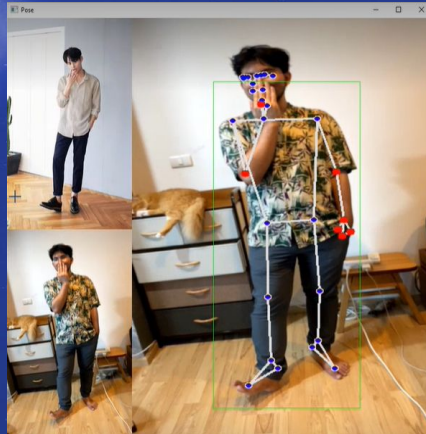
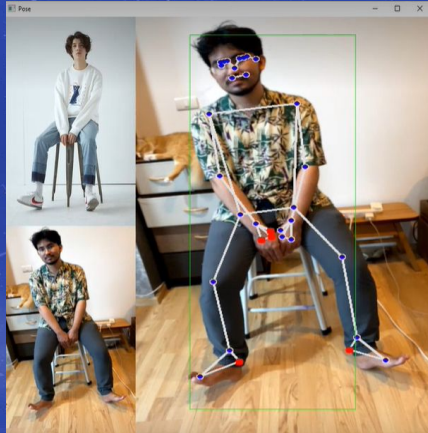


Sit



Process

Step 4 : Pose like a model



Process

Step 5 : Take pictures and Save image automatically

Example



Male ► Sit down



Male ► Stand up

02 Literature Review

2.1 Human Pose Estimation

Vivek Anand Thoutam et al.

ใช้เทคนิค Microsoft Kinect จับ Body Contour และสร้าง Body Map เพื่อสร้างตำแหน่งของคนในการออกกำลังกายโยคะ ซึ่งให้ค่า Accuracy 99.33% แต่ยากในด้านการใช้งานจริง เพราะต้องมีตัว Depth Sensor Camera

Valentin et al.

ศึกษา BlazePose พบว่าสามารถตรวจจับจุดสำคัญของร่างกายได้เพิ่มขึ้นถึง 33 จุดสำหรับบุคคลคนเดียว และความเร็วในการจับมากกว่า 30 เฟรมต่อวินาที ซึ่งเหมาะกับการใช้งานแบบ Real-Time และสามารถทำงานบน CPU มือถือได้

Zhe Cao et al.

ใช้ Deep Learning ตัว CNN ด้วยระบบ Open-source Real-Time ในการตรวจจับท่าทาง 2D แบบหลายคน และยังตรวจจับจุดสำคัญของร่างกายเช่น เท้า มือ ใบหน้า แต่ผลการทดสอบพบว่า การทับซ้อนกันของร่างกายบางส่วนทำให้ประสิทธิภาพของอัลกอริทึมลดลง จึงใช้ BlazePose ในการตรวจจับจุดสำคัญของร่างกายคน 18 จุดเพื่อแก้ปัญหาเหล่านี้

Pramote & Nalin

เสนอวิธีการนับจำนวนบุคคลเข้าหรือออกจากสถานที่ และบอกตำแหน่งของบุคคลแบบ Real-time ด้วยอัลกอริทึม YOLO โดยสามารถจำแนกวัตถุว่าเป็นชนิดใด

Arya Patki et al.

นำ BlazePose มาใช้ในการประมาณท่าทางในการตรวจจับติดตามการเคลื่อนไหวของคนในขณะออกกำลังกาย กิจกรรมโยคะ เพื่อสร้างเอาต์พุตแจ้งให้ผู้ใช้แก้ไขท่าทางเมื่อทำการออกกำลังกายหรือโยคะผิดวิธี

Kristian et al.

ศึกษา Tiny YOLO V3 และเปรียบเทียบกับอัลกอริทึมอื่นๆ ซึ่ง YOLO V3 เป็นอัลกอริทึมที่ใช้ในการตรวจจับวัตถุที่มีความเร็ว และความแม่นยำสูง และมีความเร็วมากกว่า fast RCNN SSD และ YOLO

Laklouka & Cherfi

นำ BlazePose มาใช้พัฒนาเพื่อแปลภาษามือให้เป็นคำลายลักษณ์อักษร เพื่ออำนวยความสะดวกในการสื่อสารระหว่างคนหูหนวก ใช้กับบุคคลปกติโดยไม่ต้องเรียนภาษามือ

Shahriar et al.

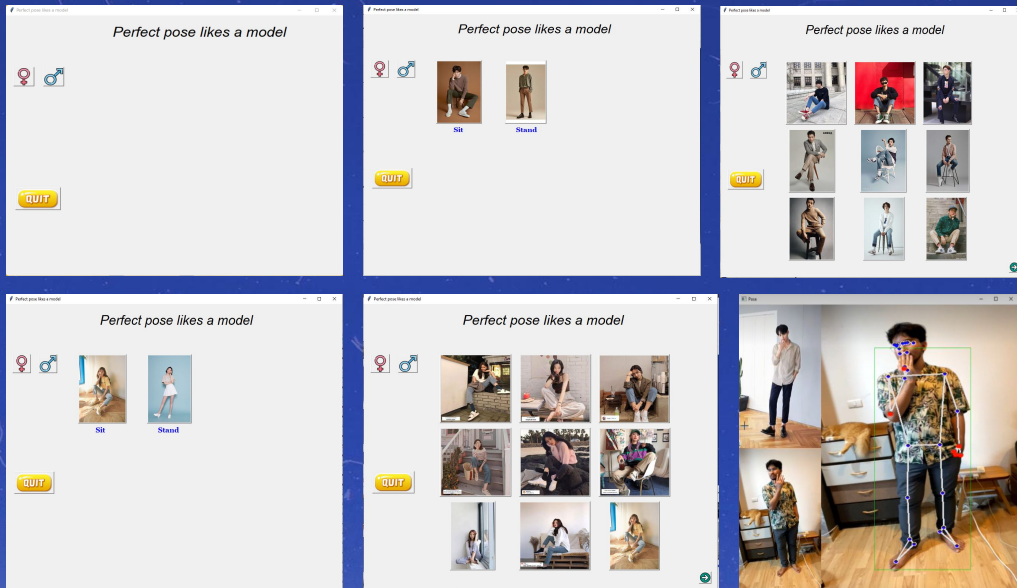
เสนอการพัฒนาเทคโนโลยีในการตรวจจับวัตถุ เช่น คนบนทางเท้า รถยนต์ หรือจักรยาน

2.2 Object Detection

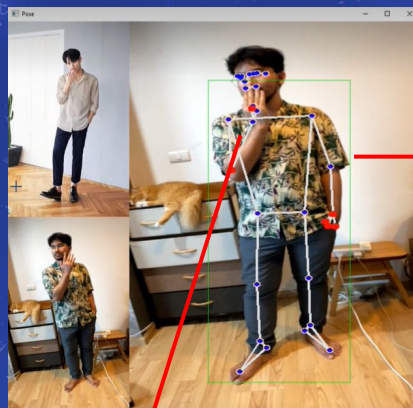
03 Research Methodology

1. Create GUI

Tkinter



2. Algorithm



Tiny YOLO V3
Algorithm

Person/pose Detection
Model
(BlazePose Detector)



03 Research Methodology

2. Agorithm

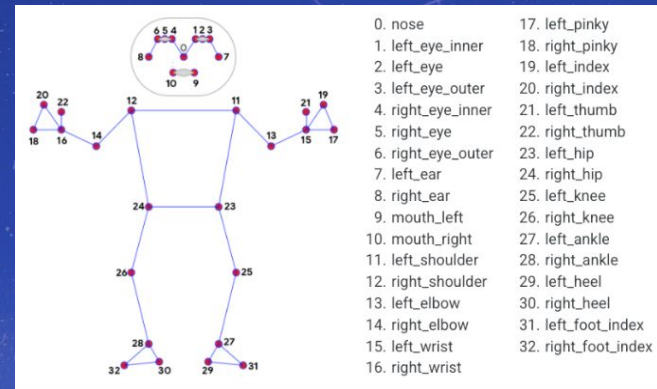
Tiny YOLO V3 Algorithm

- YOLO มีความเร็ว และความแม่นยำสูง มาตรฐานจับคน (Object Detection)
- สามารถระบุตำแหน่งในรูปภาพ เพื่อใช้ในการสร้างกรอบในการแนะนำตำแหน่งที่ควรจะมีท่าเพื่อให้ได้ภาพที่เป็นไปตามต้องการทั้งท่าทาง และตำแหน่ง

3. Model

Person/pose Detection Model (BlazePose Detector)

- อัลกอริทึม BlazePose สามารถตรวจจับจุดสำคัญที่ละเอียด ครบถ้วนทั้งร่างกาย มี 33 จุด
- มีการทำงานที่รวดเร็วซึ่งเหมาะกับการทำงานแบบ Real-time



Person/pose Detection Model (BlazePose Detector)

Step 1 Get pose landmark from mark image



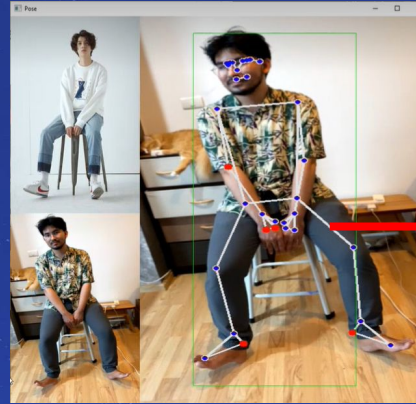
landmask_mark

```
mpPose = mp.solutions.pose
pose = mpPose.Pose()
mpDraw = mp.solutions.drawing_utils
points = mpPose.PoseLandmark # Landmarks
data = []
for p in points:
    x = str(p)[13:]
    data.append(x + "_x")
    data.append(x + "_y")
    data.append(x + "_z")
    data.append(x + "_vis")
data = pd.DataFrame(columns = data) # Empty dataset

imgRGB_mask = cv2.cvtColor(image_mask, cv2.COLOR_BGR2RGB)
results_mask = pose.process(imgRGB_mask)
lm = results_mask.pose_landmarks

landmarks_mask = results_mask.pose_landmarks.landmark
```

Step 2 Get pose landmark real time and Check similar of landmark



landmask

Check similar
with
landmask_mask



Person/pose Detection Model (BlazePose Detector)

```
i = 0
temp = []
while True:
    count = 0

    # success, img = cap.read()
    imgRGB = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    blackie = np.zeros(frame.shape) # Blank image
    newImage = frame.copy()
    results = pose.process(imgRGB)
    lm_0 = results.pose_landmarks

    # print(results.pose_landmarks)
    if results.pose_landmarks:
        # mpDraw.draw_landmarks(blackie, results.pose_landmarks, mpPose.POSE_CONNECTIONS)
        mpDraw.draw_landmarks(newImage, results.pose_landmarks, mpPose.POSE_CONNECTIONS)
        landmarks = results.pose_landmarks.landmark

        count_mistake = 0
        for id, lm in enumerate(landmarks):
            h, w, c = frame.shape
            # print(id, lm)

            temp = [landmarks[id].x, landmarks[id].y, landmarks[id].z, landmarks[id].visibility]
            # print(len(temp))

            list_A = [landmarks[id].x, landmarks[id].y, landmarks[id].z, landmarks[id].visibility]
            # print(list_A)
            # print(type(list_A))
            list_B = [landmarks_mask[id].x, landmarks_mask[id].y, landmarks_mask[id].z, landmarks_mask[id].visibility]

            cx, cy = int(landmarks[id].x*w), int(landmarks[id].y*h)
            # print(landmarks[id].visibility - landmarks_mask[id].visibility)

            if (abs(landmarks[id].visibility - landmarks_mask[id].visibility)*100) <= 0.3 :
                # print(abs(landmarks[id].visibility - landmarks_mask[id].visibility)*100)
                cv2.circle(newImage, (cx, cy), 5, (255,0,0), cv2.FILLED)
            else :
                count_mistake += 1
                cv2.circle(newImage, (cx, cy), 5, (0,0,255), cv2.FILLED)

        if count_mistake <= 5:
            dir_list = os.listdir(path_save)
            print('yes')
            cv2.imwrite(path_save + '\\img_frame_'+str(len(dir_list)+1)+'.png', frame)
            data.loc[count] = temp
            count += 1
```

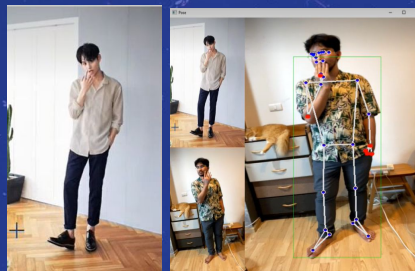
ถ้าค่า visibility ของ landmark และ landmark_mask ต่างกัน <= 0.3 จะแสดงเป็นจุดสีน้ำเงิน

ถ้าค่า visibility ของ landmark และ landmark_mask ต่างกัน > 0.3 จะแสดงเป็นจุดสีแดง และ นับเป็นจุดที่ผิด

ถ้า นับเป็นจุดที่ผิด ของ landmark แล้ว <= 5 จุด จะทำการถ่ายรูปนั้นๆไว้

04 Result and Show practical application

Image Experimental Results



WORLD PHOTO DAY



WORLD PHOTO DAY

Video Experimental Results

Click for watching Video

