嵌入式系統總整與實作

曾煜棋、吳昆儒

National Yang Ming Chiao Tung University

日期	主題
2/21	0. 課程介紹
2/28	梅竹賽!!
3/7	1. 嵌入式開發板 - 樹莓派介紹與設定 (headless)
3/14	2. 連接感測器 (GPIO, I2C)
3/21	3. 整合感測資訊 (IMU)
3/28	4. 整合音訊資料 (麥克風, 語音識別)
4/4	清明節放假
4/11	5. 整合視覺資料 (攝影機, 影像辨識)
4/18	期中考Midterm, Project分組
4/25	6. 嵌入式模型 (mediapipe, video, audio, text)
5/2	7. 喚醒詞原理 (by 台灣樹莓派)
5/9	Final Project – Proposal 分組報告 (online?)
5/16	8. 樹莓派核心編譯 (Cross compile, Kernel)
5/23	9. 嵌入式套件編譯
5/30	端午節放假
6/6	Final project準備周, Q&A (學期考試周)
6/13	**Final Project demonstration**

期中考周
(4/7-4/11)

期末考周
(6/2-6/6)

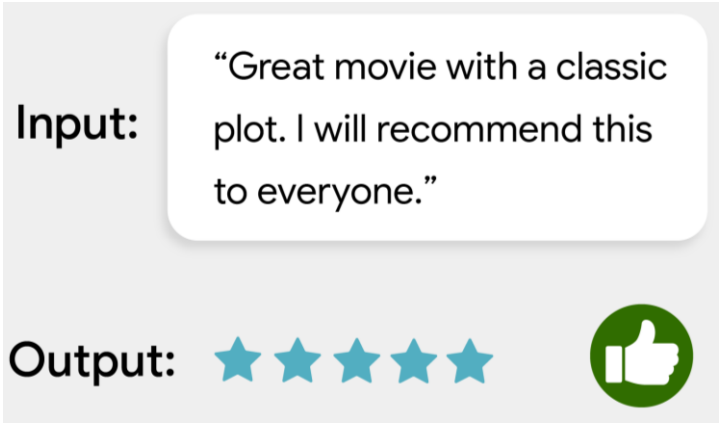
Last week

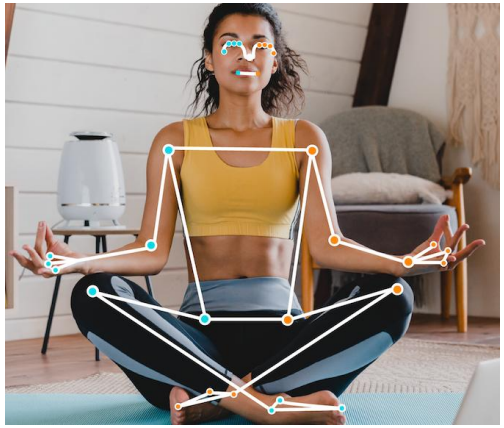
- 整合視覺資料 (攝影機, 影像辨識)
  - Part1: 安裝相機模組
  - Part2: 人臉偵測, 輪廓偵測
  - Part3: 物件辨識 (tensorflow, YOLO)

- Install dependency
  - sudo apt install python3-opencv
  - pip install imutils
  - pip install dlib

This week

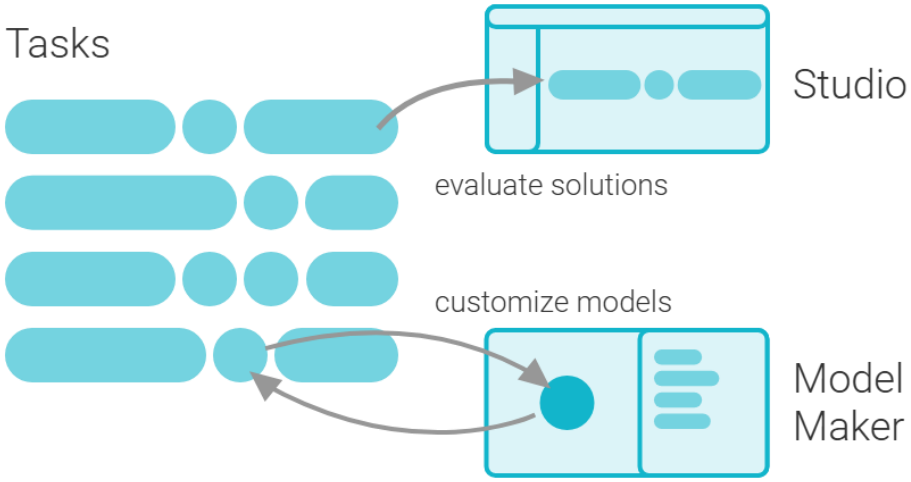
- 嵌入式模型 (mediapipe)
  - Part1: Mediapipe intro
  - Part2: Text classification
  - Part3: Audio classification
  - Part4: Image applications (Pose estimation)





MediaPipe Solutions

- MediaPipe Solutions offers a suite of AI/ML libraries and tools
- Easily and quickly integrates into various applications
- Supports multiple development platforms



Available solutions

- LLM Inference API
- Object detection
- Image classification
- Image segmentation
- Interactive segmentation
- Hand landmark detection
- Gesture recognition
- Image embedding
- Face detection
- Face landmark detection
- Face stylization
- Pose landmark detection
- Image generation
- Text classification
- Text embedding
- Language detector
- Audio classification

MediaPipe

- **Text Classification**
  - Enter text in the command line and classify it as either positive or negative with a provided confidence score
- **Audio classifier**
  - perform real-time audio classification using audio streamed from the microphone.
- **Image Application**
  - Including Face detection, Object detection, Pose landmark detection … etc

注意OS版本

- Note: Before you begin, you need to set up your Raspberry Pi with Raspberry 64-bit Pi OS.

需重新準備OS

- 下載最新版64bit OS
  - https://downloads.raspberrypi.com/raspios_arm64/images/raspios_arm64-2024-11-19/2024-11-19-raspios-bookworm-arm64.img.xz
- 設定headless參數 (cmdline.txt, config.txt)
- 建立使用者 (userconf.txt)
- 開啟SSH服務 (filename with SSH)

本周五會提供新的SD卡給大家用
(已安裝OS與mediapipe + headless設定)

Python環境注意事項

- **Starting in Raspberry Pi OS Bookworm, packages installed via pip must be installed into a Python virtual environment (venv).**
- A virtual environment is a container where you can safely install third-party modules so they won't interfere with your system Python.

直接安裝會有錯誤訊息

```
pi@raspberrypi:~ $ pip install mediapipe
error: externally-managed-environment

× This environment is externally managed
╰─> To install Python packages system-wide, try apt install
    python3-xyz, where xyz is the package you are trying to
    install.

    If you wish to install a non-Debian-packaged Python package,
    create a virtual environment using python3 -m venv path/to/venv.
    Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
    sure you have python3-full installed.

    For more information visit http://rptl.io/venv

note: If you believe this is a mistake, please contact your Python ins
tallation or OS distribution provider. You can override this, at the r
isk of breaking your Python installation or OS, by passing --break-sys
tem-packages.
hint: See PEP 668 for the detailed specification.
```

Virtualenv

- virtualenv is a tool to create isolated Python environments

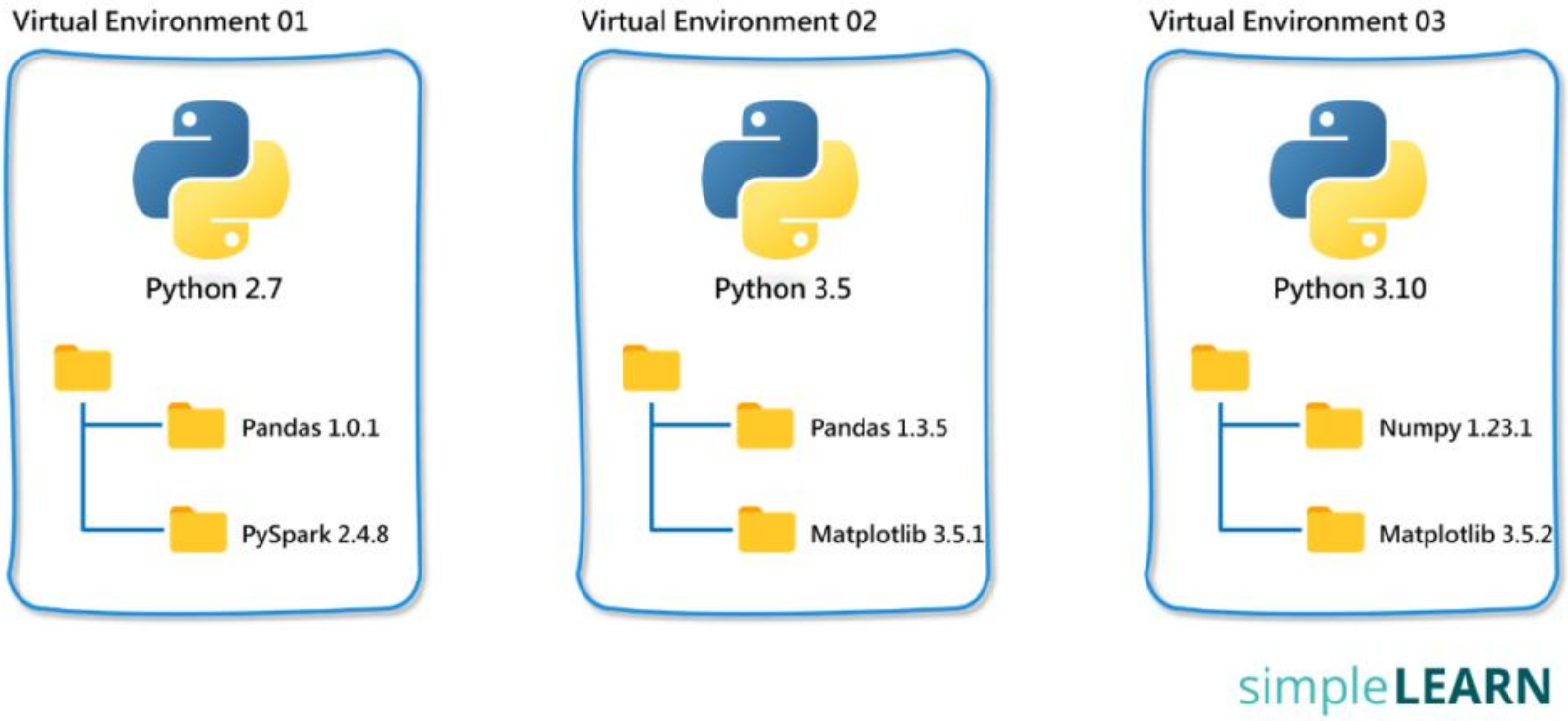


Fig source: https://simplelearn.tw/2022/08/08/ python-虛擬環境完整教學-1｜virtualenv-虛擬環境安裝及使用/

Mediapipe venv

- sudo apt-get update
- sudo apt-get install -y python3-dev python3-pip
- python -m venv --system-site-packages mdp
- **source ~/mdp/bin/activate　　// 進入虛擬環境**
- pip install matplotlib opencv-python imutils
- pip install mediapipe
- git clone https://github.com/google-ai-edge/mediapipe-samples
- ……

- **deactivate　　// 離開虛擬環境**

Text Classification

https://github.com/google-ai-edge/mediapipe-samples/tree/main/examples/text_classification/raspberry_pi

Text Classification

- This model will accept text entered in the command line and classify it as either **positive** or **negative** with a provided confidence score.
- The models were trained on the SST-2 (Stanford Sentiment Treebank) dataset, which consists of movie reviews labeled as either positive or negative.
- The supported classification models include Average Word-Embedding and MobileBERT.

Classification models

- **BERT-classifier model**
  - Based on MobileBERT; offers high accuracy
  - Includes metadata for BERT tokenization
  - Recommended for scenarios needing higher accuracy
- **Average word embedding model**
  - Uses average word embedding; smaller and faster
  - Lower prediction accuracy, but faster to retrain
  - Includes metadata for regex tokenization
  - Suitable for resource-limited or efficiency-focused scenarios

MobileBERT: a Compact Task-Agnostic BERT for Resource-Limited Devices
https://arxiv.org/abs/2004.02984

Model Name	CPU Latency
Average word embedding	0.14ms
BERT-classifier	57.68ms

Run the example

- git clone https://github.com/google-ai-edge/mediapipe-samples
- cd mediapipe-samples/examples/text_classification/raspberry_pi
- sh setup.sh
- python3 classify.py --inputText "Your text goes here"

```
(mdp) pi@raspberrypi:~/mediapipe-samples/examples/text_classification/raspberry_pi $ python3 classify
.py --inputText "Today is a wonderful day to build something people love"
Error in cpuinfo: prctl(PR_SVE_GET_VL) failed
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
W0000 00:00:1745167556.943222    5747 inference_feedback_manager.cc:114] Feedback manager requires a
model with a single signature inference. Disabling support for feedback tensors.
positive (1.00)
```

Discussion 1

- The sample code use BERT-classifier model by default.
- How to use **average word embedding model** to classify the text?

https://docs.python.org/3/library/subprocess.html

Quiz 1

- Integrate Microphone Input, Speech-to-Text, and Text Classification
- Capture real-time audio from a microphone, transcribe the speech to text, and then classify the resulting text using a text classification model
- You have to rebuild environment
  - pip install SpeechRecognition
  - pip install openai
- Hint: you can use python's **subprocess**

```
Please wait. Calibrating microphone...
Say something!
OpenAI Whisper API thinks you said: The quick fox jumps over a lazy dog.
classify.py stdout:
 negative (0.64)
```

Speech-to-Text example

```python
import speech_recognition as sr
import os

#obtain audio from the microphone
r=sr.Recognizer()

with sr.Microphone() as source:
    print("Please wait. Calibrating microphone...")
    #listen for 1 seconds and create the ambient noise energy level
    r.adjust_for_ambient_noise(source, duration=1)
    print("Say something!")
    audio=r.listen(source)

# recognize speech using Whisper API
OPENAI_API_KEY = "    "
os.environ["OPENAI_API_KEY"] = OPENAI_API_KEY
try:
    print(f"OpenAI Whisper API thinks you said {r.recognize_openai(audio)}")
except sr.RequestError as e:
    print(f"Could not request results from OpenAI Whisper API; {e}")

```

Audio classifier

https://github.com/google-ai-edge/mediapipe-samples/tree/main/examples/audio_classifier/raspberry_pi

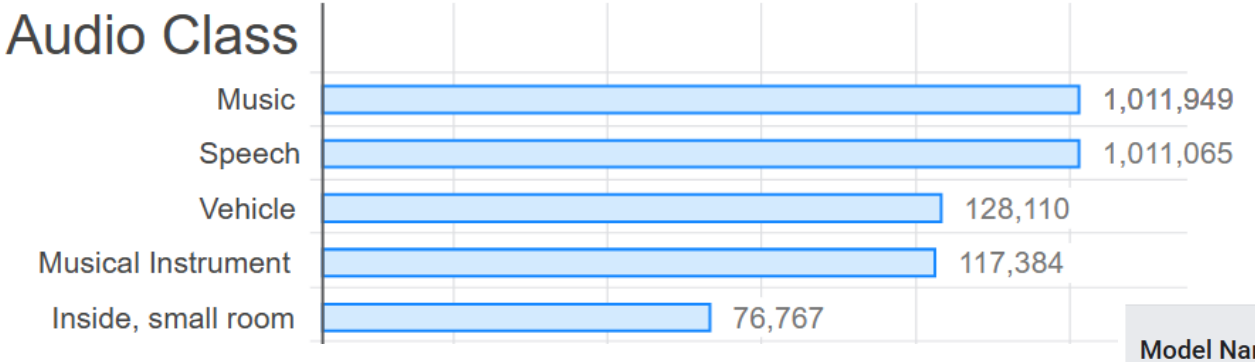
Audio classification

- This model **classifies audio clips** into a set of **defined categories**, such as guitar music, a train whistle, or a bird's song.
- The categories are defined during the training of the model.
- This task operates on audio data with a machine learning (ML) model as independent audio clips or a continuous stream and outputs a list of potential categories ranked by descending probability score.

Classification models

- **Yamnet model**
  - This audio event classifier trained on the AudioSet dataset to predict audio events defined in the AudioSet data.
- **AudioSet dataset**
  - A large-scale dataset of manually annotated audio events

Audio Set: An ontology and human-labeled dataset for audio events
https://ieeexplore.ieee.org/document/7952261

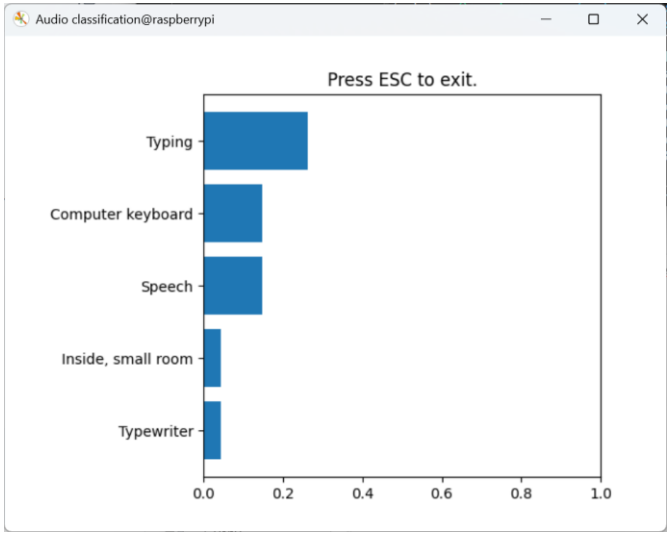


Model Name	CPU Latency
YamNet	12.29ms

Run the example

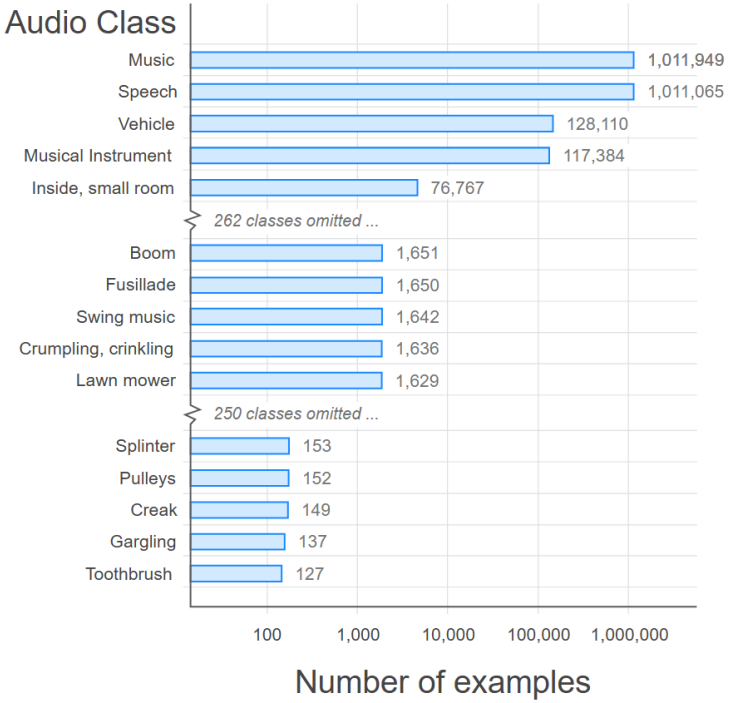
- git clone https://github.com/google-ai-edge/mediapipe-samples
- cd mediapipe/examples/audio_classifier/raspberry_pi
- sh setup.sh
- python3 classify.py

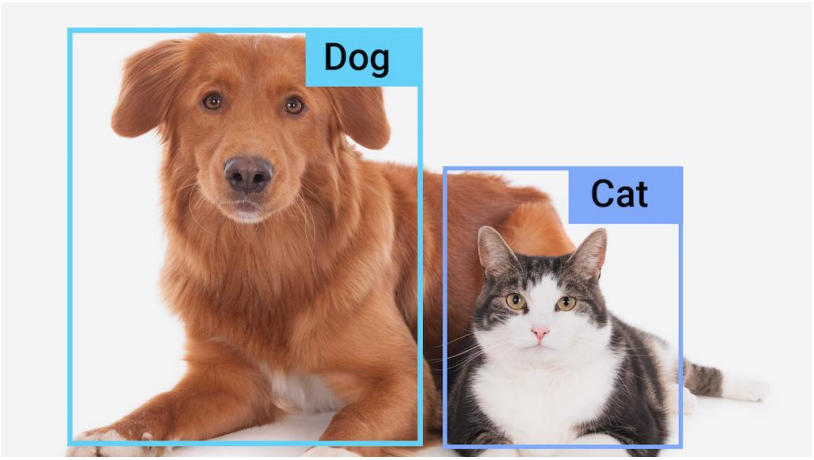
```
(mdp) pi@raspberrypi:~/mediapipe-samples/examples/audio_classifier/raspberry_pi $ python3 classify.py
Error in cpuinfo: prctl(PR_SVE_GET_VL) failed
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
W0000 00:00:1745205146.017366    6428 inference_feedback_manager.cc:114] Feedback manager requires a mo
del with a single signature inference. Disabling support for feedback tensors.
W0000 00:00:1745205147.092885    6428 time_series_util.cc:52] Timestamp 1745205147.090000 not consisten
t with number of samples 15600 and initial timestamp 1745205146601000.  Expected timestamp: 1745205147.
576000 Timestamp difference: -0.486000 sample_rate: 16000.000000
W0000 00:00:1745205159.628734    6429 time_series_util.cc:52] Timestamp 1745205159.628000 not consisten
t with number of samples 327600 and initial timestamp 1745205146601000.  Expected timestamp: 174520516
.076000 Timestamp difference: -7.448000 sample_rate: 16000.000000
```

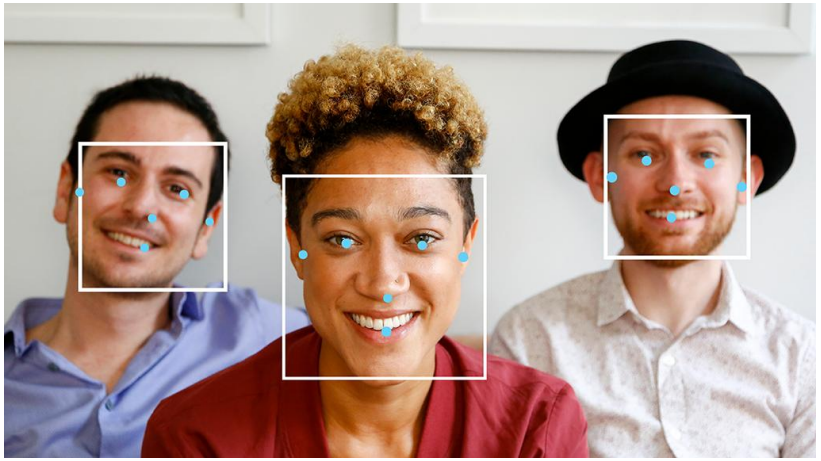


Discussion 2

- AudioSet consists of an expanding ontology of 632 audio event classes and a collection of 2,084,320 human-labeled 10-second sound clips drawn from YouTube videos.
- How can we find the **full list** of audio events?







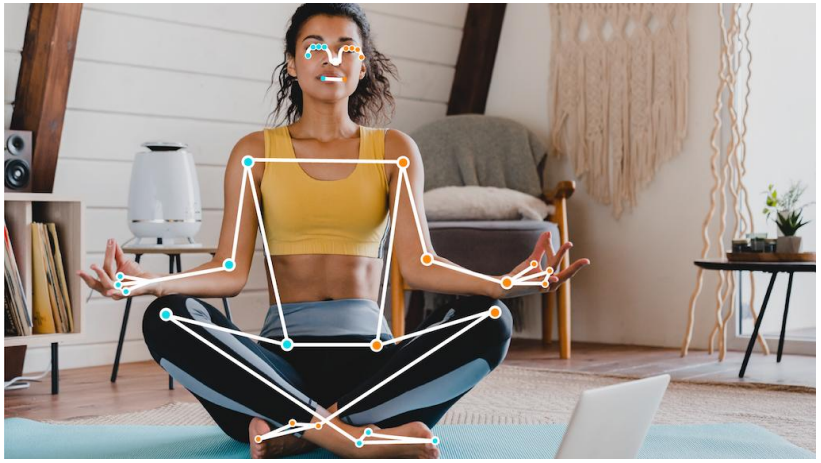
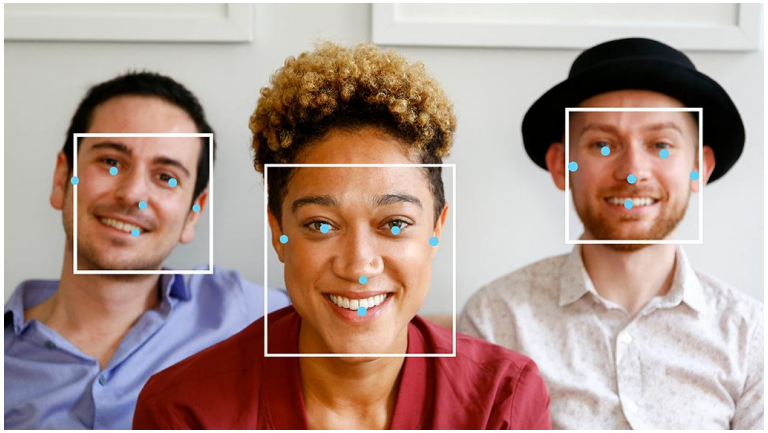


Image Application

https://github.com/google-ai-edge/mediapipe-samples/tree/main/examples/face_detector/raspberry_pi
https://github.com/google-ai-edge/mediapipe-samples/tree/main/examples/object_detection/raspberry_pi
https://github.com/google-ai-edge/mediapipe-samples/tree/main/examples/pose_landmarker/raspberry_pi

Face detection

- This model **detects faces** in an image or video.
- You can use this task to locate faces and facial features within a frame.
- The task outputs **face locations**, along with the following **facial key points**: left eye, right eye, nose tip, mouth, left eye tragion, and right eye tragion.



Models

- The models are variants of **BlazeFace**, a lightweight and accurate face detector optimized for mobile GPU inference.
- BlazeFace models are suitable for applications like 3D facial keypoint estimation, expression classification, and face region segmentation.

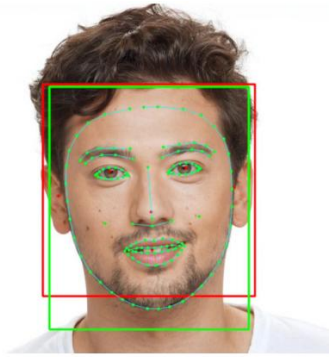


Figure 3. Pipeline example (best viewed in color). Red: BlazeFace output. Green: Task-specific model output.

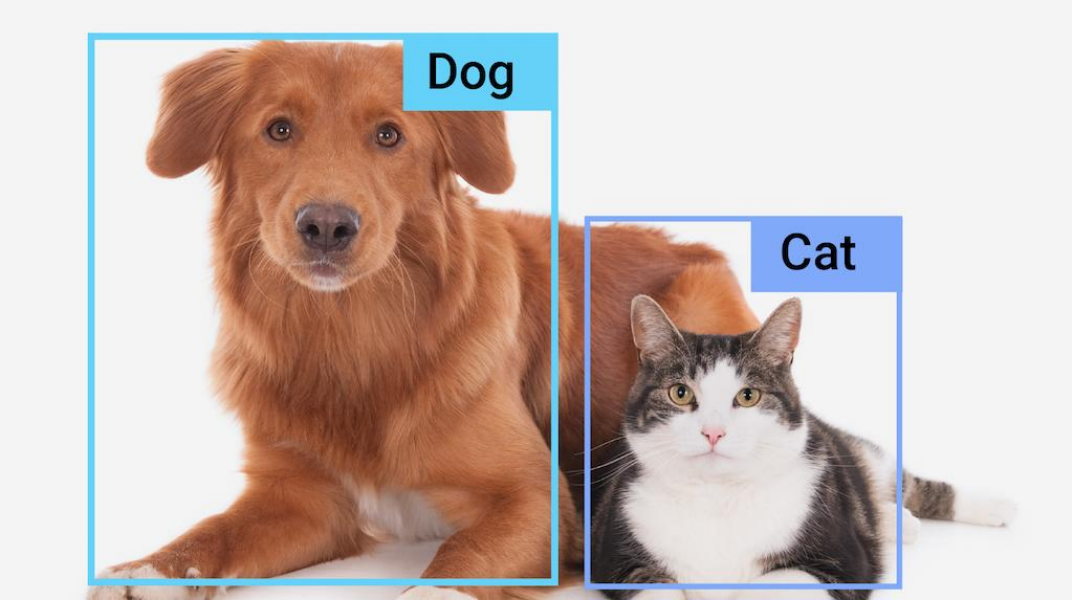
BlazeFace: Sub-millisecond Neural Face Detection on Mobile GPUs
https://arxiv.org/pdf/1907.05047

Comparisons

- **Haar Cascade**
  - Uses hand-crafted Haar features and combines many weak classifiers with AdaBoost.
  - Sensitive to lighting and pose changes, leading to more false detections.
- **HOG (Histogram of Oriented Gradients)**
  - Extracts edge orientation features and classifies using an SVM.
  - Less robust to pose, lighting, and background variations; not ideal for real-time.
- **BlazeFace**
  - Uses a lightweight CNN for fast and accurate real-time face detection.
  - Well-suited for mobile devices and can handle faces from multiple angles.

Object detection

- This model detects the presence and location of multiple classes of objects within images or videos.
- Each detection result represents an object that appears within the image or video.



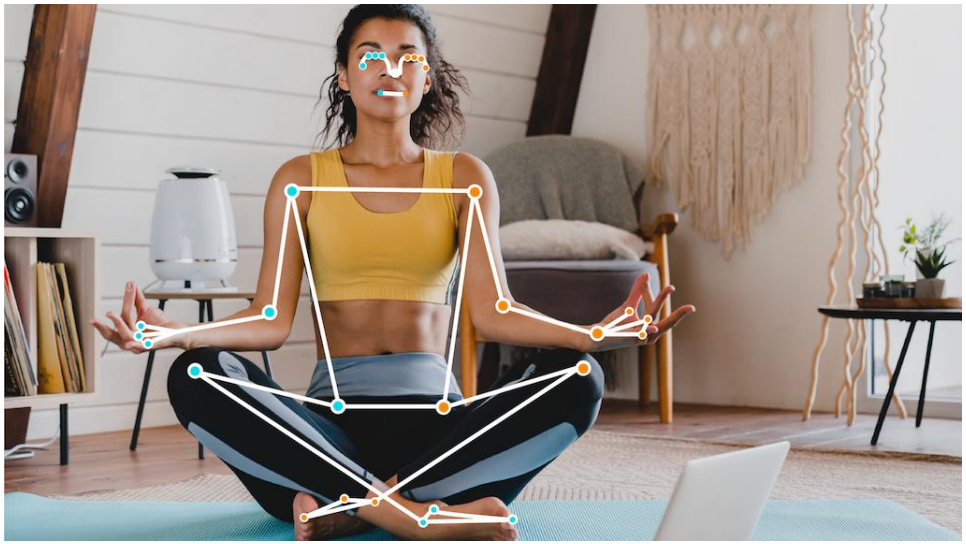
Models

- **EfficientDet-Lite0 (Recommended)**
  - Balanced in accuracy and speed, suitable for many lightweight use cases.
  - Uses a 320x320 input with EfficientNet-Lite0 backbone; supports int8/float16/float32.
- **EfficientDet-Lite2**
  - Achieves higher accuracy than Lite0 but is slower and uses more memory.
  - Uses a 448x448 input with EfficientNet-Lite2 backbone; suitable when accuracy is the top priority.
- **SSD MobileNetV2**
  - Fastest and lightest among the three models but less accurate.
  - Uses a 256x256 input with MobileNetV2 backbone; ideal when speed and small size are critical.

EfficientDet: Scalable and Efficient Object Detection
https://openaccess.thecvf.com/content_CVPR_2020/html/Tan_EfficientDet_Scalable_and_Efficient_Object_Detection_CVPR_2020_paper.html

Pose landmark detection

- The MediaPipe Pose Landmarker task lets you **detect landmarks of human bodies** in an image or video.
- You can use this task to identify **key body locations**, analyze posture, and categorize movements.
- The task outputs body pose landmarks in image coordinates and in 3-dimensional world coordinates.



Models

- The Pose Landmarker uses a series of models to predict pose landmarks.
  - **Pose detection model**: detect the presence of bodies with a few key pose landmarks.
  - **Pose landmarker model**: add a complete mapping of the pose. The model outputs an estimate of 33 3-dimensional pose landmarks.

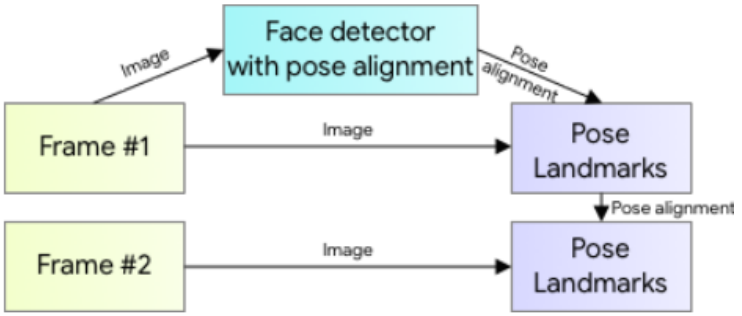
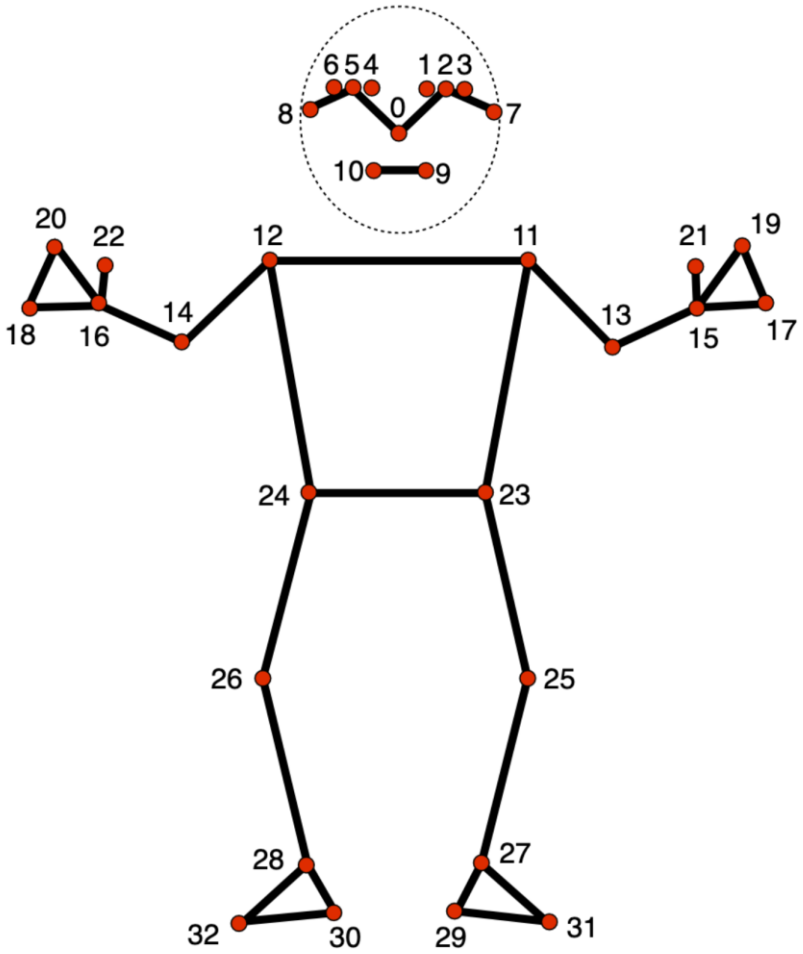


Figure 1. Inference pipeline. See text.

BlazePose: On-device Real-time Body Pose tracking
https://arxiv.org/pdf/2006.10204

Pose landmarker model



0 - nose
1 - left eye (inner)
2 - left eye
3 - left eye (outer)
4 - right eye (inner)
5 - right eye
6 - right eye (outer)
7 - left ear
8 - right ear
9 - mouth (left)
10 - mouth (right)
11 - left shoulder
12 - right shoulder
13 - left elbow
14 - right elbow
15 - left wrist
16 - right wrist
17 - left pinky
18 - right pinky
19 - left index
20 - right index
21 - left thumb
22 - right thumb
23 - left hip
24 - right hip
25 - left knee
26 - right knee
27 - left ankle
28 - right ankle
29 - left heel
30 - right heel
31 - left foot index
32 - right foot index

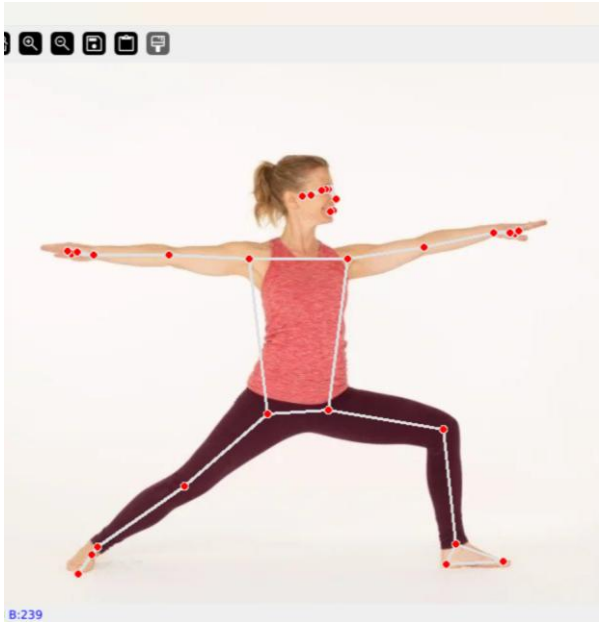
Run the example

- git clone https://github.com/google-ai-edge/mediapipe-samples
- cd mediapipe/examples/pose_landmarker/raspberry_pi
- sh setup.sh
- python3 detect.py

- Note: This example uses MediaPipe with Python on a Raspberry Pi to perform real-time pose landmarks detection using images streamed from the camera.

Run sample code

- mp_image.py: read a image to calculate keypoints
- Usage: python mp_image.py your_image



```
(mdp) pi@raspberrypi:~/mediapipe-samples/examples/pose_landmarker/raspberry_pi $ python mp_image.py war
rior-pose.jpg
Downloading model to /home/pi/mdp/lib/python3.11/site-packages/mediapipe/modules/pose_landmark/pose_lan
dmark_heavy.tflite
Error in cpuinfo: prctl(PR_SVE_GET_VL) failed
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
WARNING: All log messages before absl::InitializeLog() is called are written to STDERR
W0000 00:00:1745220846.891218    7364 inference_feedback_manager.cc:114] Feedback manager requires a mo
del with a single signature inference. Disabling support for feedback tensors.
W0000 00:00:1745220847.230398    7367 inference_feedback_manager.cc:114] Feedback manager requires a mo
del with a single signature inference. Disabling support for feedback tensors.
W0000 00:00:1745220847.771356    7367 landmark_projection_calculator.cc:186] Using NORM_RECT without IM
AGE_DIMENSIONS is only supported for the square ROI. Provide IMAGE_DIMENSIONS or use PROJECTION_MATRIX.
======================
0 NOSE (534, 141, -107.0648655295372, 0.9999992847442627)
======================
1 LEFT_EYE_INNER (528, 131, -88.58142793178558, 0.9999954700469971)
======================
2 LEFT_EYE (527, 131, -88.80846947431564, 0.9999960660934448)
======================
3 LEFT_EYE_OUTER (526, 131, -88.7698233127594, 0.9999959468841553)
======================
4 RIGHT_EYE_INNER (526, 131, -112.45574802160263, 0.999997615814209)
======================
5 RIGHT_EYE (523, 131, -112.72476613521576, 0.9999983310699463)
======================
6 RIGHT_EYE_OUTER (519, 132, -112.77415603399277, 0.9999980926513672)
======================
7 LEFT_EAR (508, 137, -7.074025459587574, 0.9999982118606567)
======================
```

載入套件與初始化模型

```
#!/usr/bin/python3
import cv2
import sys
import time
import numpy as np
import mediapipe as mp

mp_drawing = mp.solutions.drawing_utils
mp_pose = mp.solutions.pose
pose = mp_pose.Pose(static_image_mode=True, min_detection_confidence=0.3, model_complexity=2)
```

讀取圖片

```
try:
    image = cv2.imread(sys.argv[1])
except Exception as e:
    print(e)
    image = cv2.imread("images/warrior-pose.jpg")
```

姿態辨識

```
output_image = image.copy()
imageRGB = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
results = pose.process(imageRGB)
height, width, _ = image.shape
landmarks = []
```

輸出座標(x,y,z)

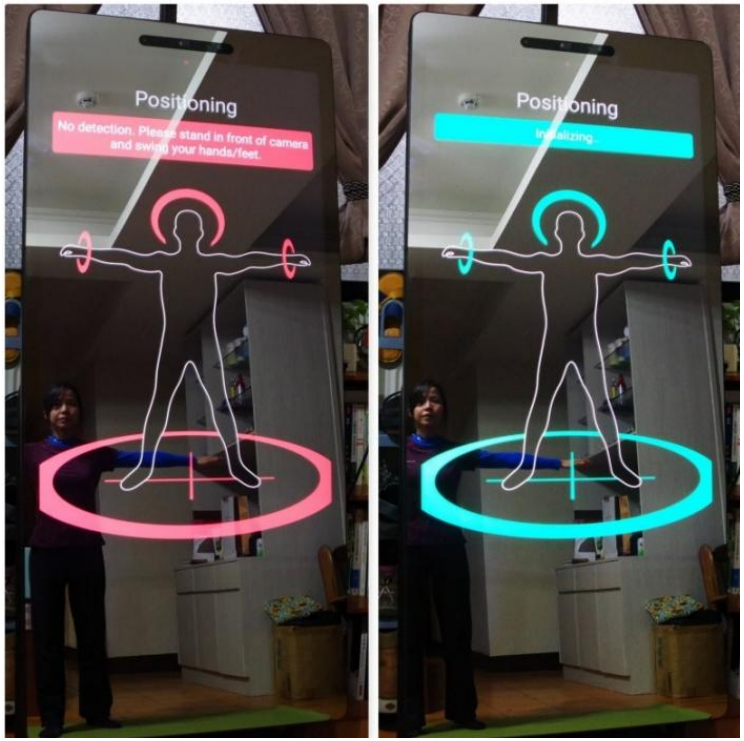
```
# Check if any landmarks are detected.
if results.pose_landmarks:
    # Draw Pose landmarks on the output image.
    mp_drawing.draw_landmarks(image=output_image, landmark_list=results.pose_landmarks,
        connections=mp_pose.POSE_CONNECTIONS)
    # Iterate over the detected landmarks.
    for landmark in results.pose_landmarks.landmark:
        # Append the landmark into the list.
        landmarks.append((int(landmark.x * width), int(landmark.y * height),
            (landmark.z * width), landmark.visibility))

for idx, landmark in enumerate(landmarks):
    print("======================")
    print(idx, mp_pose.PoseLandmark(idx).name, landmark)
```

顯示結果

```
cv2.imshow("preview", output_image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

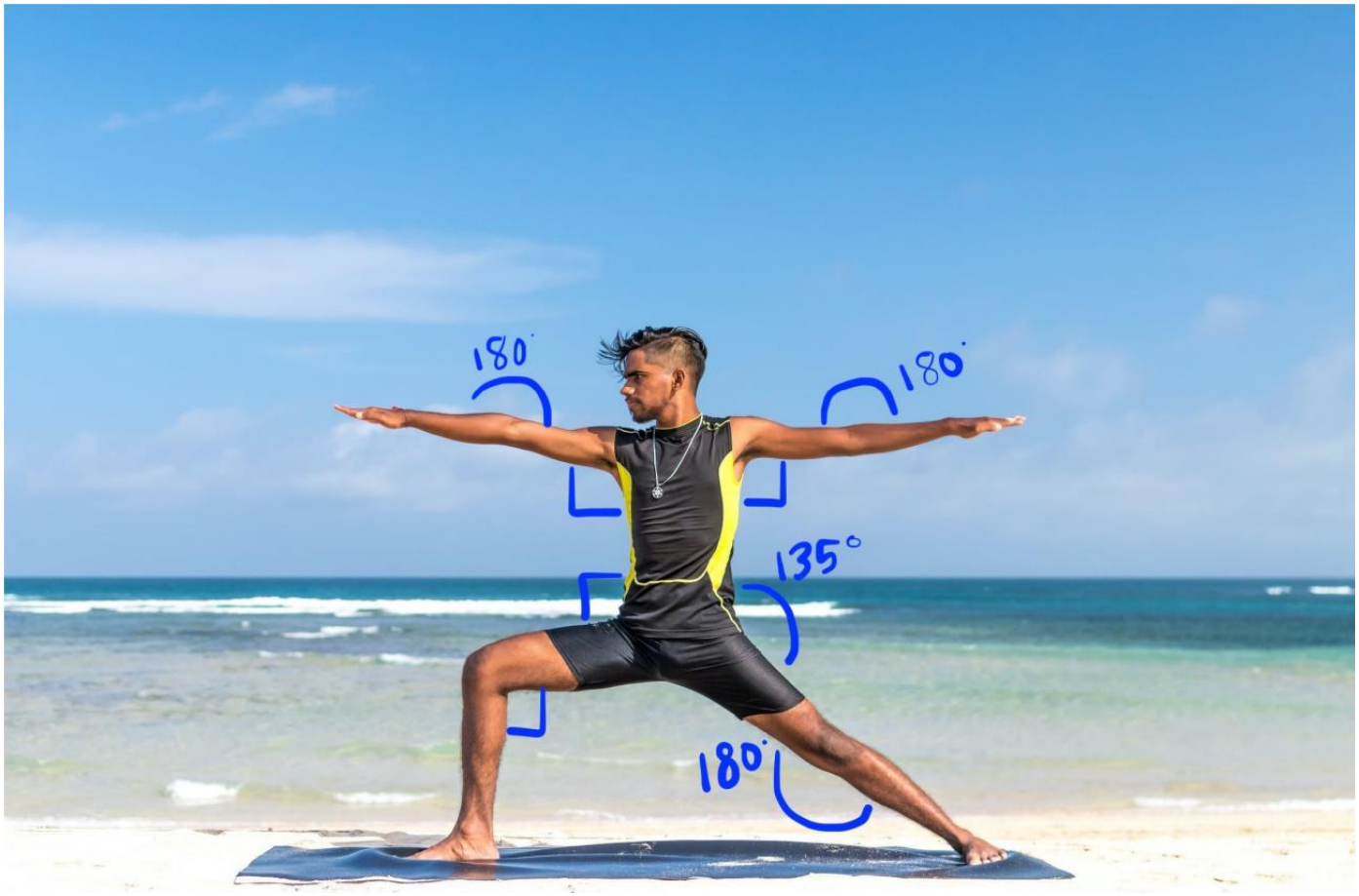
姿態識別應用

- APP: 健身比對姿勢, 復健治療, 運動強化追蹤



https://www.luxurywatcher.com/zh-Hant/article/29928

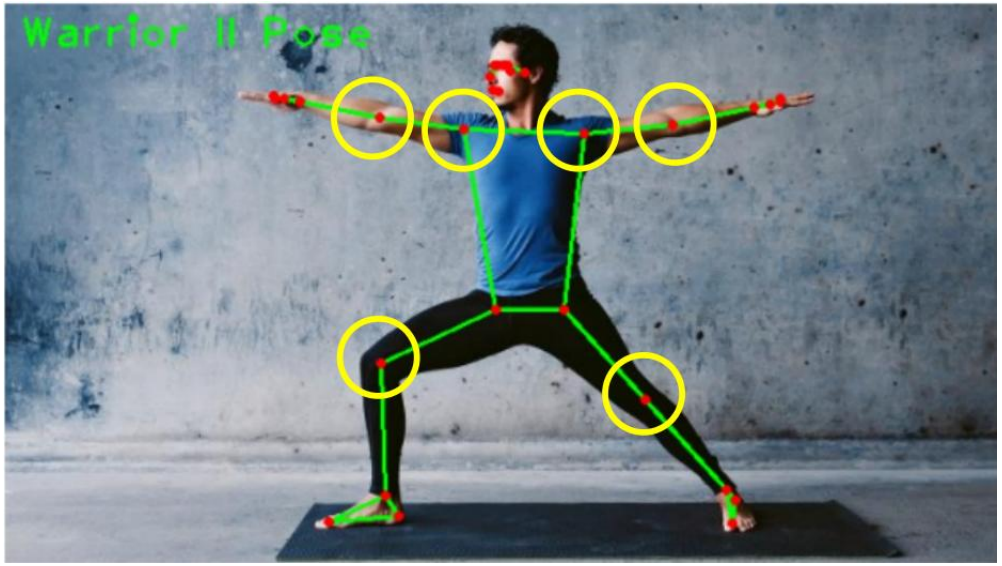
姿態分類基礎



https://developers.google.com/ml-kit/vision/pose-detection/classifying-poses

姿態分類

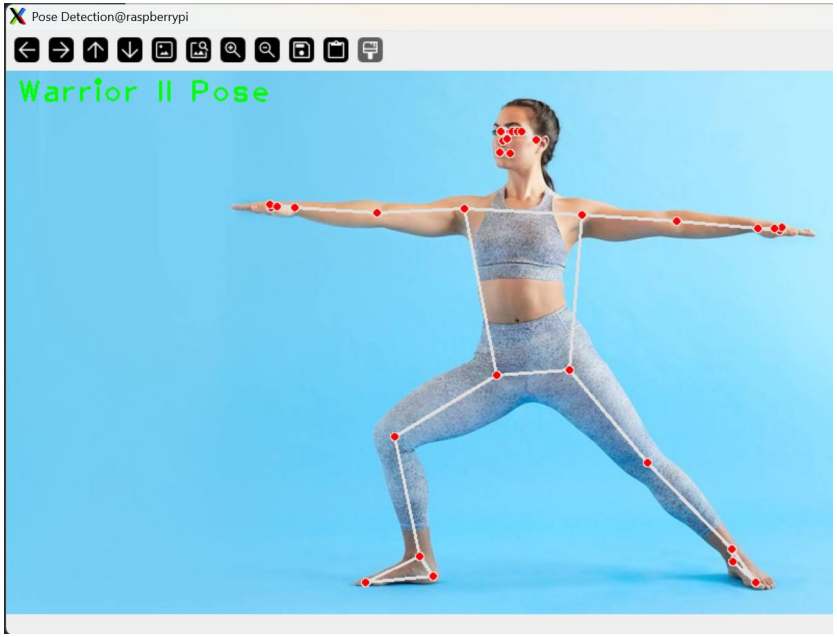
- 計算多個關鍵點間的角度 + 條件判斷
- 以Warrior II Pose為例
  - 195 > left_elbow_angle > 165
  - 110 > left_shoulder_angle > 80
  - 195 > left_knee_angle > 165
  - 120 > right_knee_angle > 90



判斷姿勢sample code

- mp_pose_compare.py: 判斷姿勢
- Usage: python mp_pose_compare.py warrior-pose-ii.jpg

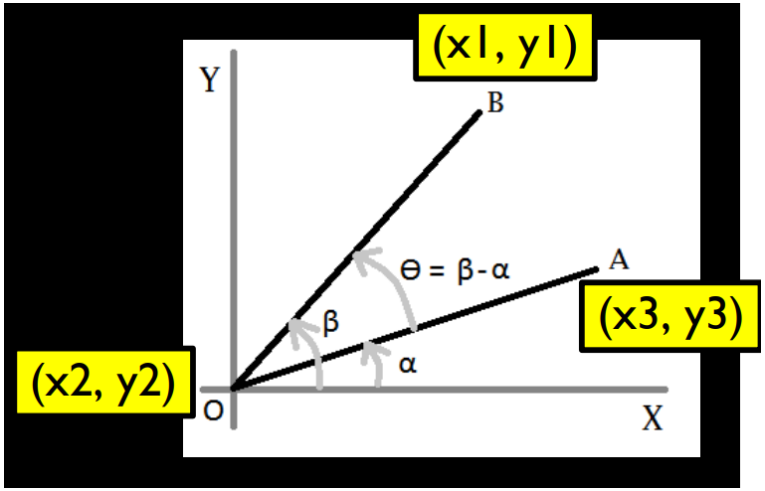
```
========================================
left_elbow_angle: 181
right_elbow_angle: 186
# 1. Both arms are straight
========================================
left_shoulder_angle: 90
right_shoulder_angle: 98
# 2. Shoulders are at the required angle
========================================
left_knee_angle: 175
right_knee_angle 109
# 3. One leg is straight
========================================
left_knee_angle: 175
right_knee_angle 109
# 4. The other leg is bended at the required angle
========================================
```



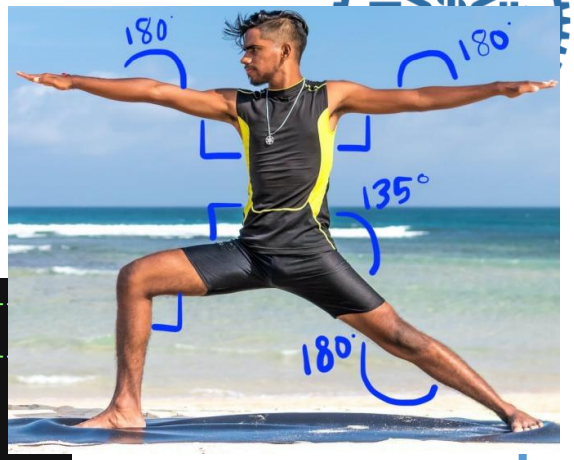
計算三個點之間的夾角

```python
def calculateAngle(landmark1, landmark2, landmark3):
    x1, y1, _ = landmark1
    x2, y2, _ = landmark2
    x3, y3, _ = landmark3

    angle = math.degrees(math.atan2(y3 - y2, x3 - x2) - math.atan2(y1 - y2, x1 - x2))

    if angle < 0:
        angle += 360

    angle = min(angle, 360 - angle)

    return angle
```

夾角等於兩個向量的arctan差值



挑選關節點



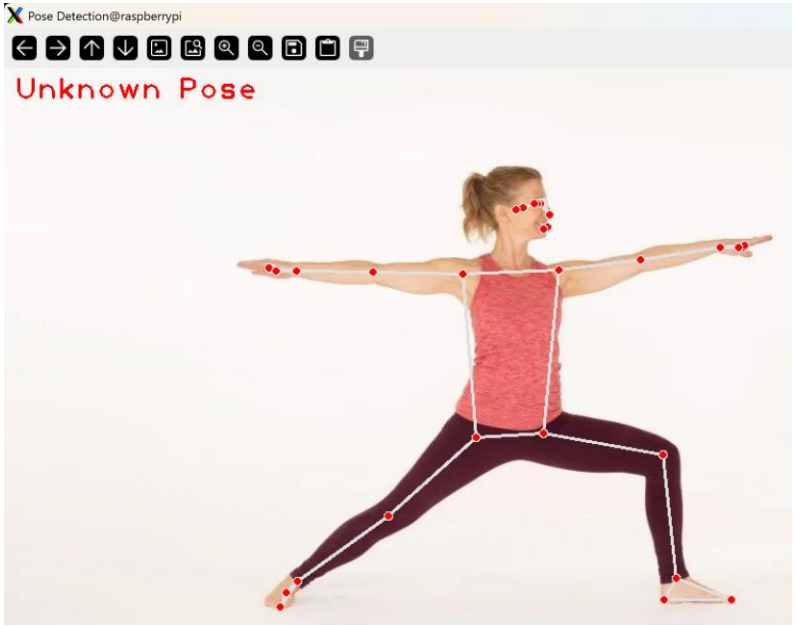

```
#--------------------------------------------------------------------------------------------------
# Calculate the required angles.
#--------------------------------------------------------------------------------------------------

try:
    # Get the angle between the left shoulder, elbow and wrist points.
    left_elbow_angle = calculateAngle(landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value],
                                      landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value],
                                      landmarks[mp_pose.PoseLandmark.LEFT_WRIST.value])

    # Get the angle between the right shoulder, elbow and wrist points.
    right_elbow_angle = calculateAngle(landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value],
                                       landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value],
                                       landmarks[mp_pose.PoseLandmark.RIGHT_WRIST.value])

    # Get the angle between the left elbow, shoulder and hip points.
    left_shoulder_angle = calculateAngle(landmarks[mp_pose.PoseLandmark.LEFT_ELBOW.value],
                                         landmarks[mp_pose.PoseLandmark.LEFT_SHOULDER.value],
                                         landmarks[mp_pose.PoseLandmark.LEFT_HIP.value])

    # Get the angle between the right hip, shoulder and elbow points.
    right_shoulder_angle = calculateAngle(landmarks[mp_pose.PoseLandmark.RIGHT_HIP.value],
                                          landmarks[mp_pose.PoseLandmark.RIGHT_SHOULDER.value],
                                          landmarks[mp_pose.PoseLandmark.RIGHT_ELBOW.value])

    # Get the angle between the left hip, knee and ankle points.
    left_knee_angle = calculateAngle(landmarks[mp_pose.PoseLandmark.LEFT_HIP.value],
                                     landmarks[mp_pose.PoseLandmark.LEFT_KNEE.value],
                                     landmarks[mp_pose.PoseLandmark.LEFT_ANKLE.value])

    # Get the angle between the right hip, knee and ankle points
    right_knee_angle = calculateAngle(landmarks[mp_pose.PoseLandmark.RIGHT_HIP.value],
                                      landmarks[mp_pose.PoseLandmark.RIGHT_KNEE.value],
                                      landmarks[mp_pose.PoseLandmark.RIGHT_ANKLE.value])
```

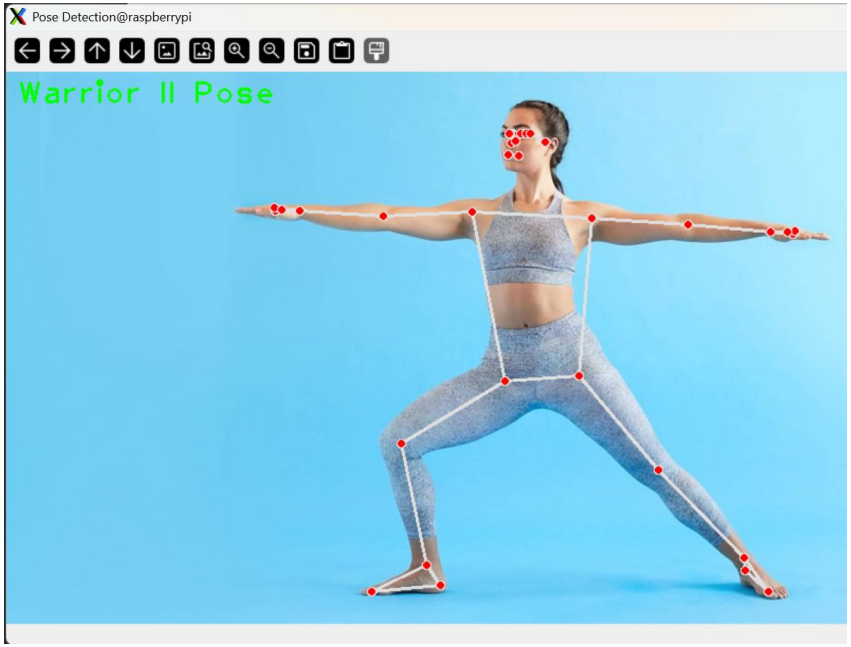
檢查姿勢

```
#----------------------------------------------------------------------------
# Check if it is the warrior II pose
#----------------------------------------------------------------------------

print("======================================")
# 1. Check if the both arms are straight.
print("left_elbow_angle:", int(left_elbow_angle))
print("right_elbow_angle:", int(right_elbow_angle))
if 165 < left_elbow_angle < 195 and 165 < right_elbow_angle < 195:
    print("# 1. Both arms are straight")
    print("======================================")

    print("left_shoulder_angle:", int(left_shoulder_angle))
    print("right_shoulder_angle:", int(right_shoulder_angle))
    # 2. Check if shoulders are at the required angle.
    if 80 < left_shoulder_angle < 110 and 80 < right_shoulder_angle < 110:
        print("# 2. Shoulders are at the required angle")
        print("======================================")

        print("left_knee_angle:", int(left_knee_angle))
         # 3. Check if one leg is straight.
        if 165 < left_knee_angle < 195:
            print("# 3. One leg is straight")
            print("======================================")

            # 4. Check if the other leg is bended at the required angle.
            print("right_knee_angle", int(right_knee_angle))
            if 90 < right_knee_angle < 120:
                print("# 4. The other leg is bended at the required angle")
                print("======================================")

                # Specify the label of the pose that is Warrior II pose.
                label = 'Warrior II Pose'
```

Quiz 2

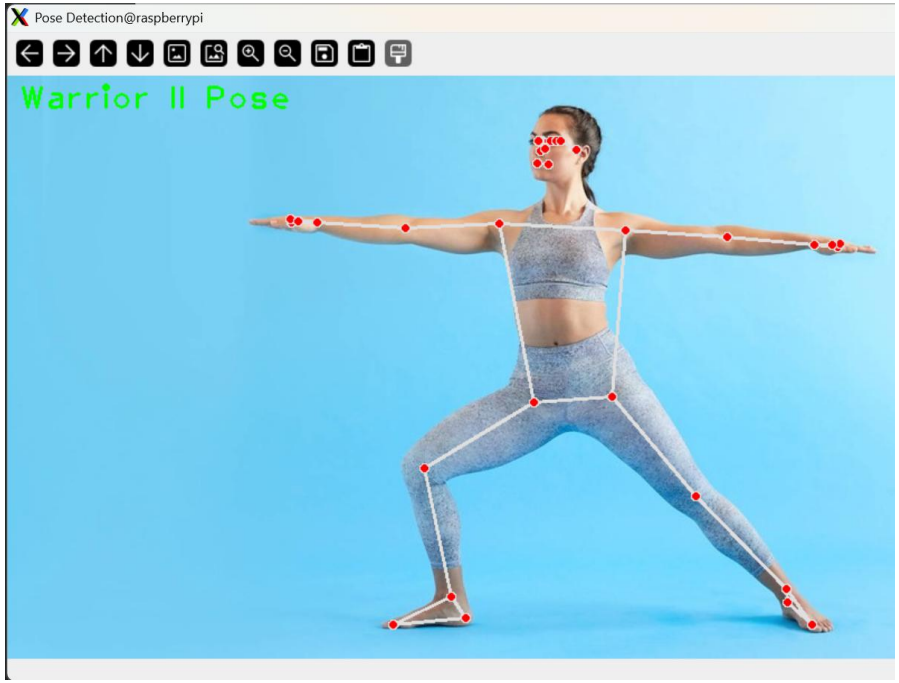
- Another pose image is also warrior II, why it shows “unknown pose”?
- Please show the correct result to TA





Discussion 3

- If your flexibility isn't good, how can you adjust the code to improve detection accuracy?



Summary

- Practice Lab: 嵌入式模型 (mediapipe, video, audio, text)

- Write down the answer for discussion 1-3
  - D1: How to use average word embedding model to classify the text?
  - D2: How can we find the full list of audio events?
  - D3: How to adjust code for better detection with limited flexibility?

- Quiz:
  - Q1: Integrate Microphone Input, Speech-to-Text, and Text Classification
  - Q2: Correct the code to capture pose