

使用人體姿態檢測模型進行脊椎檢測

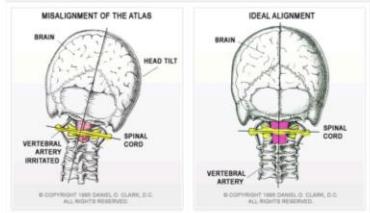
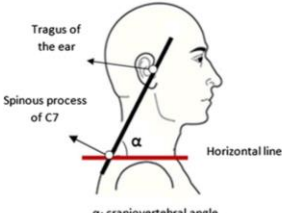
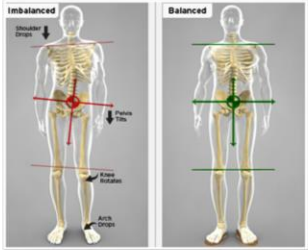
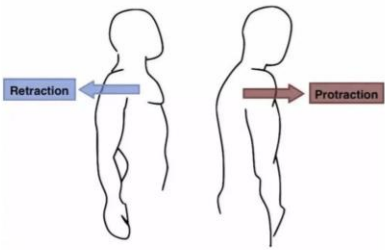
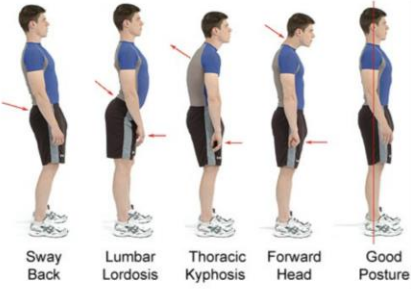
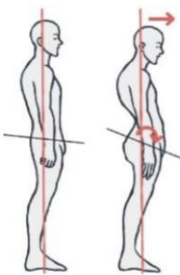
8927 AILab



- 01 前言
- 02 課程一：入門 Human Pose Estimation Model
- 03 課程二：實作 Human Pose Estimation Model
- 04 課程三：如何運用 Human Pose Estimation Model 檢測人體偏移
- 05 課程四：運用 Human Pose Estimation Model 檢測人體側身



現代醫療進步
可透過體態優化調整體態
來改善脊椎側彎、骨盆前傾的疾病
但檢測項目多樣、檢測時間較長
本案修改AI模型增加檢測關鍵點
協助檢測人員快速了解受檢者狀況
將檢測時長從1-2小時
縮短至3-5分鐘

枕骨偏移	頭骨角度計算
	
體態偏移示意圖	肩膀前後傾示意圖
	
站立姿勢評估	正常脊椎、後移脊椎對比圖
	



課程一：

入門 Human Pose Estimation Model

- 介紹[Stacked Hourglass Network]
[Regional Multi-person Pose Estimation (RMPE) framework]



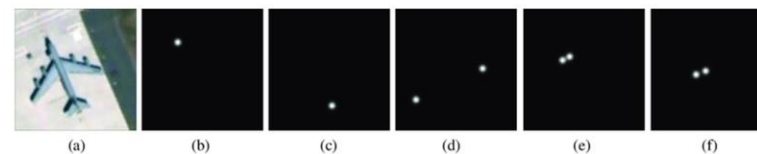
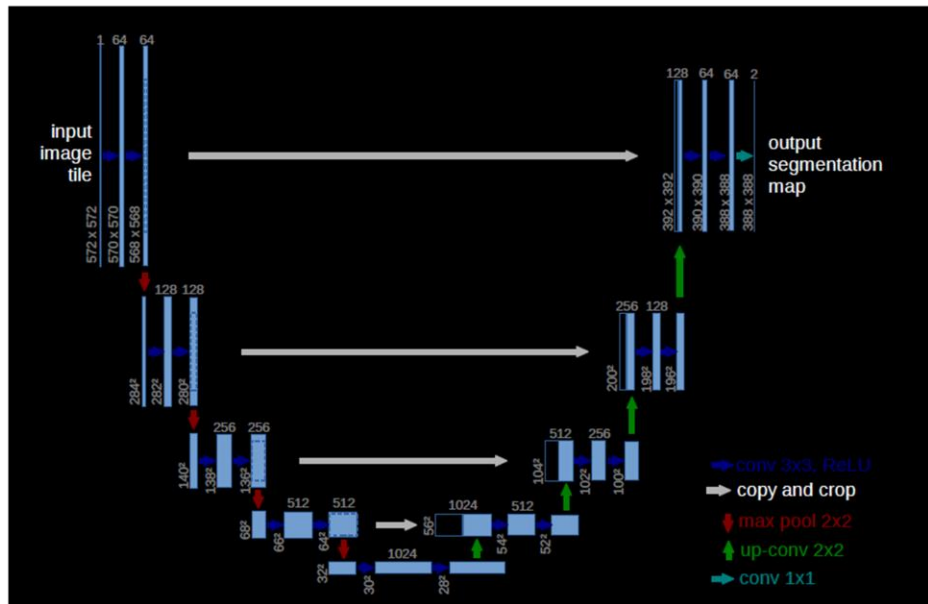
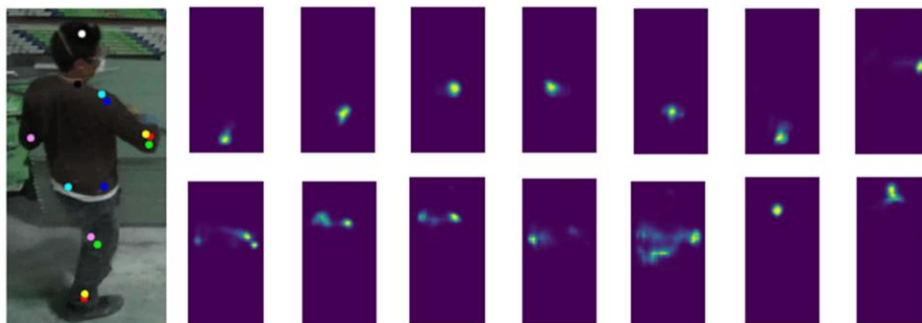
Stacked Hourglass Networks for Human Pose Estimation

Alejandro Newell, Kaiyu Yang, and Jia Deng

2016 ECCV



基礎架構與檢測結果





Related work

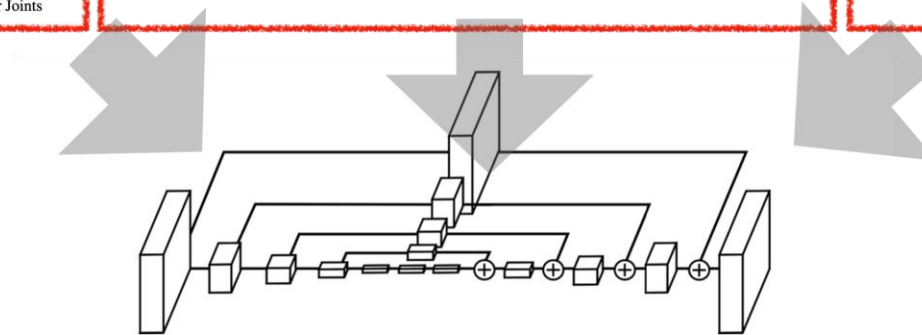
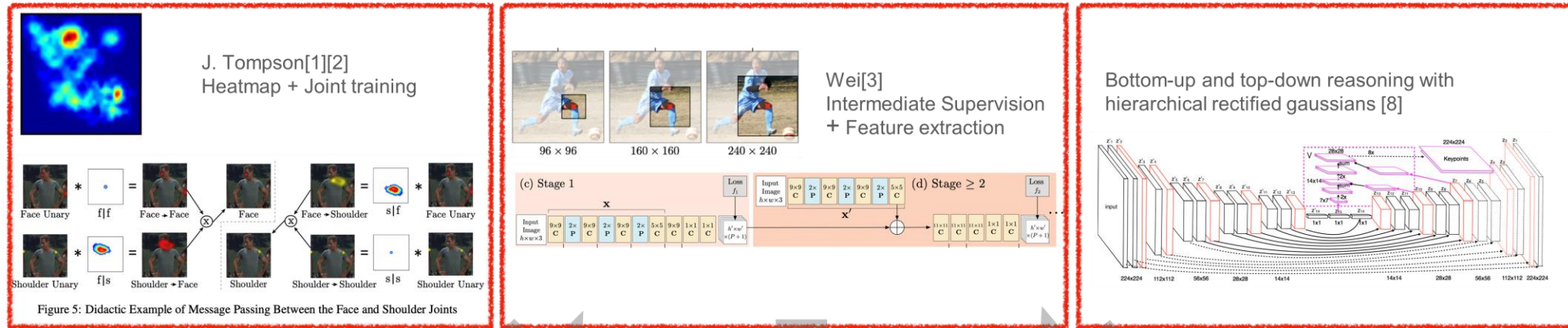


Fig. 3. An illustration of a single “hourglass” module. Each box in the figure corresponds to a residual module as seen in Figure 4. The number of features is consistent across the whole hourglass.

[1] Hu, P., Ramanan, D.: Bottom-up and top-down reasoning with hierarchical rectified gaussians. In: Computer Vision and Pattern Recognition (CVPR), 2016 .



Method

Hourglass Design

- Local evidence and coherent understanding of full body
- Skip layer : preserve spatial information
- Down/Up sample and combination of features
- 1×1 Conv : Segmentation

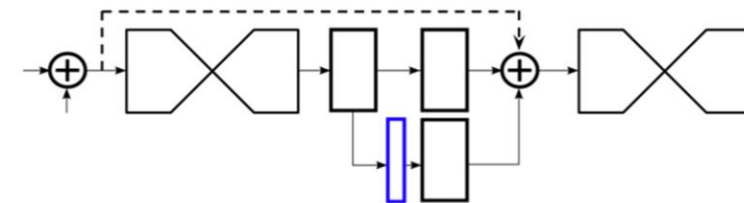
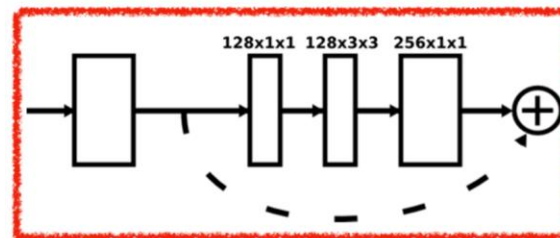


Fig. 4. Left: Residual Module [14] that we use throughout our network. **Right:** Illustration of the intermediate supervision process. The network splits and produces a set of heatmaps (outlined in blue) where a loss can be applied. A 1×1 convolution remaps the heatmaps to match the number of channels of the intermediate features. These are added together along with the features from the preceding hourglass.

Residual Unit

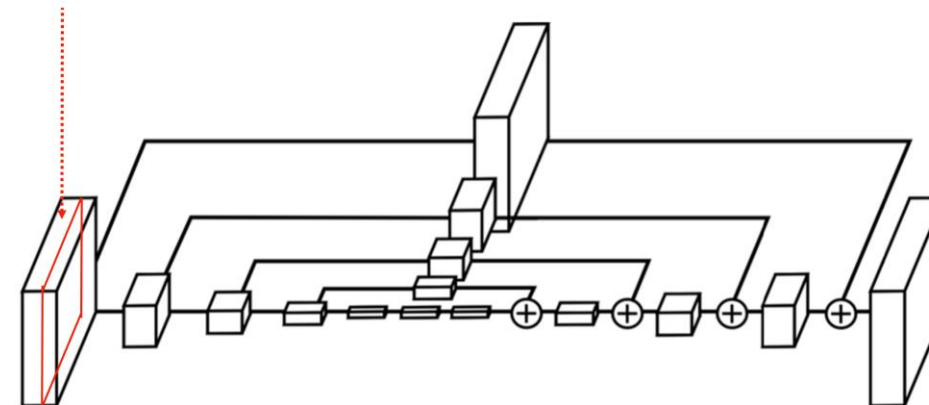


Fig. 3. An illustration of a single “hourglass” module. Each box in the figure corresponds to a residual module as seen in Figure 4. The number of features is consistent across the whole hourglass.



Ablation Study

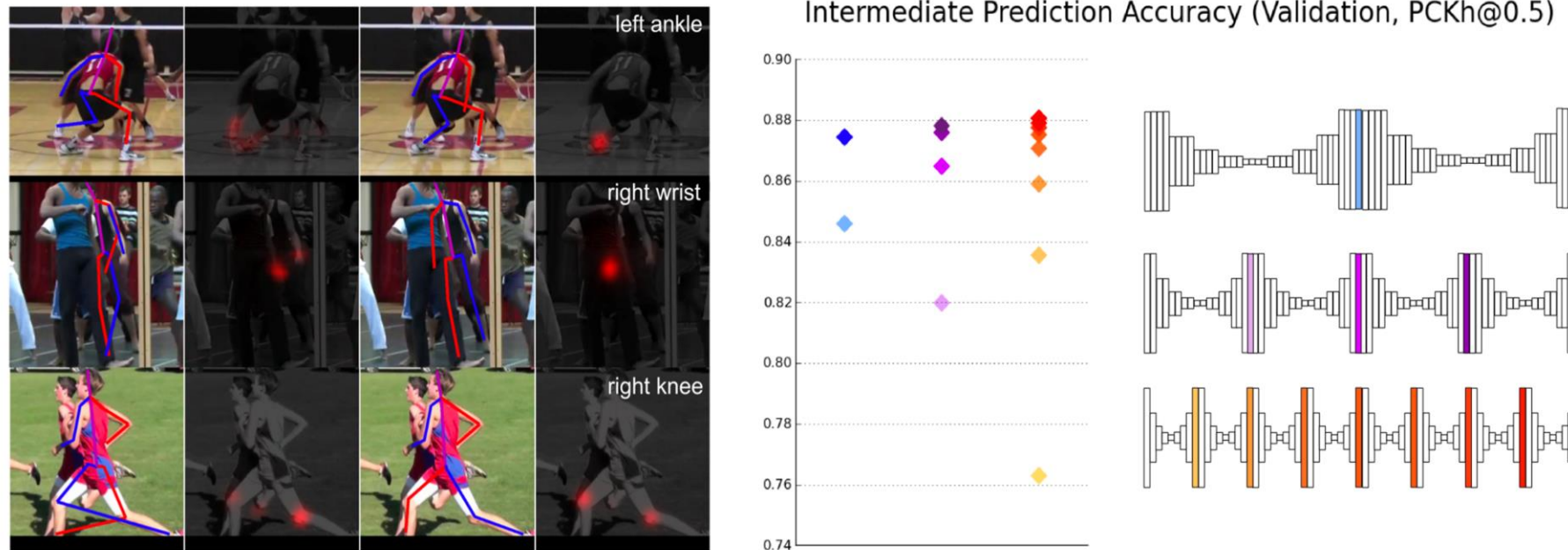


Fig. 9. Left: Example validation images illustrating the change in predictions from an intermediate stage (second hourglass) (left) to final predictions (eighth hourglass) (right). **Right:** Validation accuracy at intermediate stages of the network compared across different stacking arrangements.



Result

Fig. 11.1 Comparison on MPII

	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Total
Tompson et al. [16], CVPR'15	96.1	91.9	83.9	77.8	80.9	72.3	64.8	82.0
Carreira et al. [19], CVPR'16	95.7	91.7	81.7	72.4	82.8	73.2	66.4	81.3
Pishchulin et al. [17], CVPR'16	94.1	90.2	83.4	77.3	82.6	75.7	68.6	82.4
Hu et al. [27], CVPR'16	95.0	91.6	83.0	76.6	81.9	74.5	69.5	82.4
Wei et al. [18], CVPR'16	97.8	95.0	88.7	84.0	88.4	82.8	79.4	88.5
Our model	98.2	96.3	91.2	87.1	90.1	87.4	83.6	90.9

Table 2. Results on MPII Human Pose (PCKh@0.5)

3.1. Symmetric STN and Parallel SPPE

Human proposals provided by human detectors are not well-suited to SPPE. This is because SPPE is specifically trained on single person images and is very sensitive to localisation errors. It has been shown that small translation or cropping of human proposals can significantly affect performance of SPPE [28]. Our symmetric STN + parallel SPPE was introduced to enhance SPPE when given imperfect human proposals. The module of our SSTN and parallel SPPE is shown in Figure 4.

- [28] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *arXiv preprint arXiv:1603.06937*, 2016. [4321](#), [4322](#), [4323](#), [4326](#)



RMPE: Regional Multi-Person Pose Estimation



Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, Cewu Lu , Shanghai Jiao Tong University, China Tencent YouTu

2019 ICCV

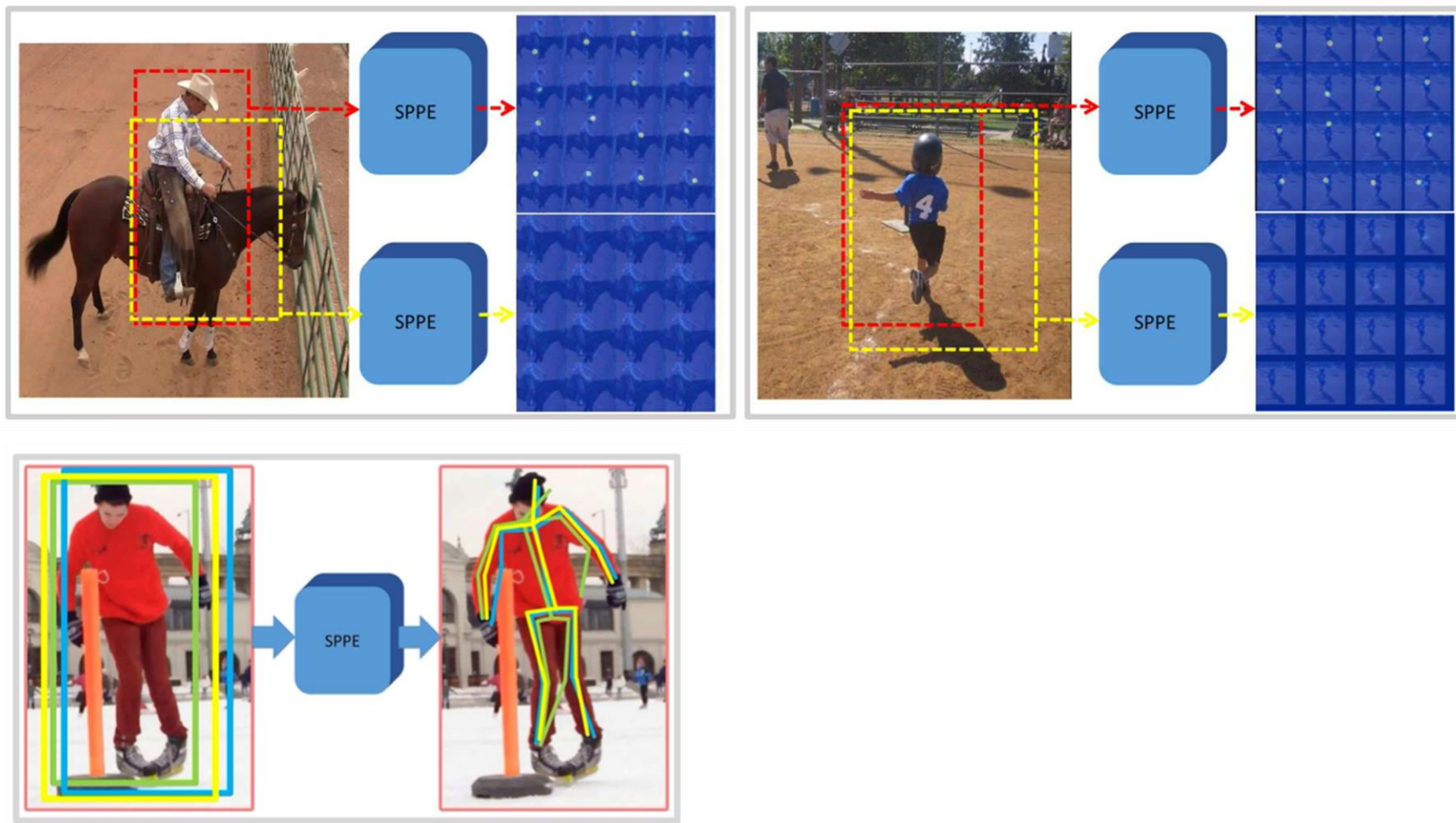


Introduction

- Part-based framework
 - Poses are ambiguous when two or more persons are too close together
 - Loses the capability to recognize body parts from a global pose view
- Two-step framework
 - Accuracy highly depends on the quality of the detected bounding boxes.
 - Flexible human detector (Faster-RCNN) & SPPE framework(SPPE Stacked Hourglass model[4])
 - Even for the cases when the bounding boxes are considered as correct with $\text{IoU} > 0.5$, the detected human poses can still be wrong.



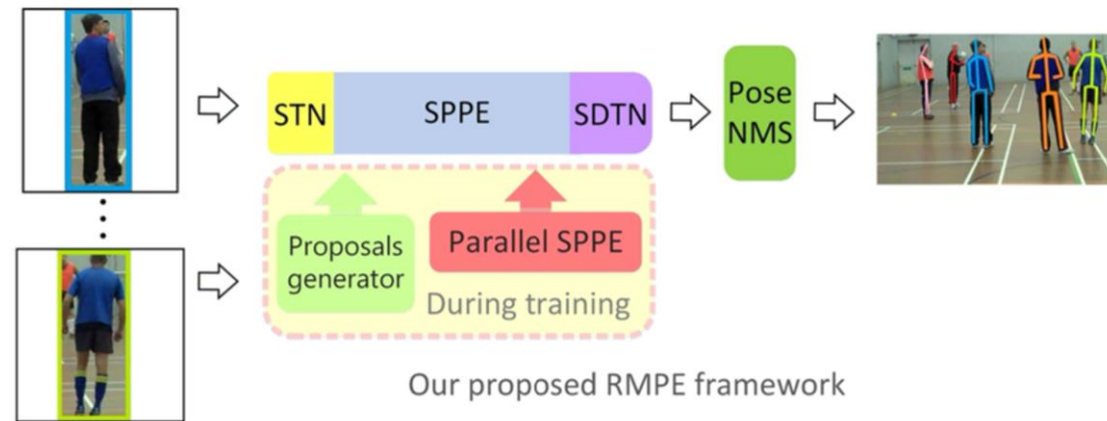
Introduction - Detect Error





Contributions and Result

- Novelty RMPE framework
- STN/SDTN
- Parallel SPPE for training STN
- Proposals Generator
- Pose NMS to avoid redundant detect
- achieves 76.7 mAP on MPII



Method	Hea	Sho	Elb	Wri	Hip	Kne	Ank	mAP	s/image
Subset of 288 images as in [1]									
Deepcut [1]	73.4	71.8	57.9	39.9	56.7	44.0	32.0	54.1	57995
Iqbal et al. [41]	70.0	65.2	56.4	46.1	52.7	47.9	44.5	54.7	10
DeeperCut [2]	87.9	84.0	71.9	63.9	68.8	63.8	58.1	71.2	230
Newell et al. [48]	91.5	87.2	75.9	65.4	72.2	67.0	62.1	74.5	-
ArtTrack [47]	92.2	91.3	80.8	71.4	79.1	72.6	67.8	79.3	0.005
Fang et al. [6]	89.3	88.1	80.7	75.5	73.7	76.7	70.0	79.1	-
Ours	92.9	91.3	82.3	72.6	76.0	70.9	66.8	79.0	0.005
Full testing set									
DeeperCut [2]	78.4	72.5	60.2	51.0	57.2	52.0	45.4	59.5	485
Iqbal et al. [41]	58.4	53.9	44.5	35.0	42.2	36.7	31.1	43.1	10
Levinko et al. [72]	89.8	85.2	71.8	59.6	71.1	63.0	53.5	70.6	-
ArtTrack [47]	88.8	87.0	75.9	64.9	74.2	68.8	60.5	74.3	0.005
Fang et al. [6]	88.4	86.5	78.6	70.4	74.4	73.0	65.8	76.7	-
Newell et al. [48]	92.1	89.3	78.9	69.8	76.2	71.6	64.7	77.5	-
Fieraru et al. [73]	91.8	89.5	80.4	69.6	77.3	71.7	65.5	78.0	-
Ours (one scale)	89.0	84.9	74.9	64.2	71.0	65.6	58.1	72.5	0.005
Ours	91.2	87.6	77.7	66.8	75.4	68.9	61.7	75.6	0.005

*From Openpose



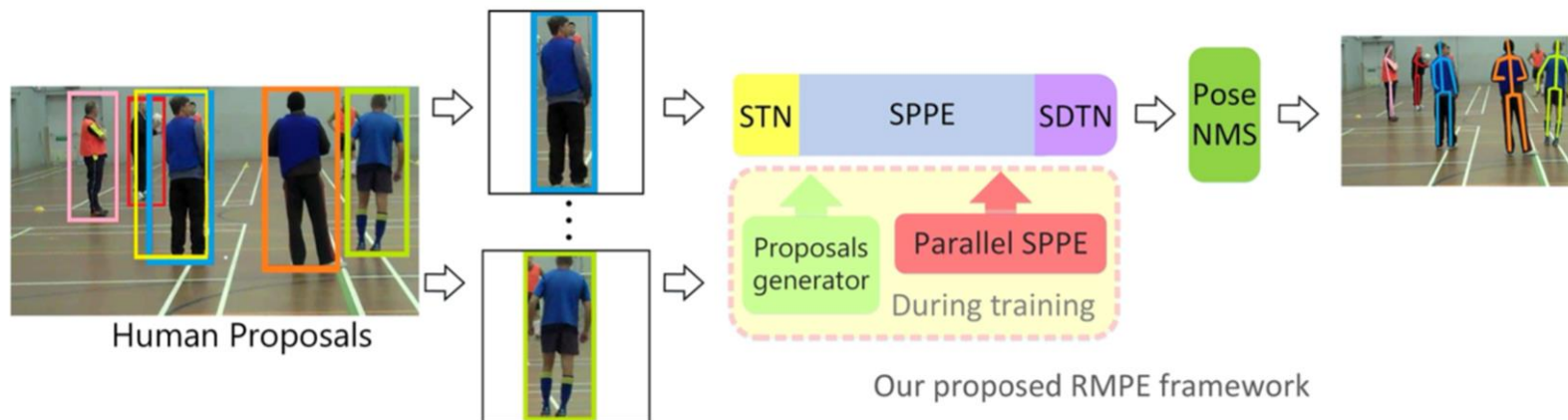
Related work

- Two-step Framework
 - SPPE method
 - We use a CNN based SPPE method to estimate poses, while Pishchulin et al [2]
 - Human detector
 - Use the Faster R-CNN as their human detector.
 - Their method can only achieve 51.0 in mAP on MPII dataset, while ours can achieve 76.7 mAP. With the development of object detection and single person pose estimation, the two-step framework can achieve further advances in its performance.



Method

- Symmetric STN + SPPE
 - Symmetric STN consists of STN and SDTN
- Parametric Pose NMS to obtain the estimated human poses
- Parallel SPPE in order to avoid local minimums and further leverage the power of SSTN
- To augment the existing training samples, a pose-guided proposals generator (PGPG) is designed.





Detection Result





Experiments

The proposed method is qualitatively and quantitatively evaluated on two standard multi-person datasets with large occlusion cases: [MPII](#) and [MSCOCO 2016 Keypoints Challenge](#) dataset.



Experiments

Team	AP	AP^{50}	AP^{75}	AP^M	AP^L
CMU-Pose[7]	61.8	84.9	67.5	57.1	68.2
G-RMI[30]	68.5	87.1	75.5	65.8	73.3
Mask R-CNN[18]	63.1	87.3	68.7	57.8	71.4
Megvii[10]	72.1	91.4	80.0	68.7	77.2
ours	61.8	83.7	69.8	58.6	67.6
ours++	72.3	89.2	79.1	68.0	78.6

Table 2. Results on the MSCOCO Keypoint Challenge (AP) dataset [2]. The MSCOCO website provides a technical overview only. Our result is obtained without ensembling. “++” denotes using faster-rcnn with softnms [5] as human detector, PyraNet [45] with input size 320x256 as pose estimator. We only compare to single model results.

	Head	Shoulder	Elbow	Wrist	Hip	Knee	Ankle	Total
full testing set								
Iqbal&Gall, ECCVw16 [41]	58.4	53.9	44.5	35.0	42.2	36.7	31.1	43.1
DeeperCut, ECCV16 [21]	78.4	72.5	60.2	51.0	57.2	52.0	45.4	59.5
Levinkov <i>et al.</i> , CVPR17[13]	89.8	85.2	71.8	59.6	71.1	63.0	53.5	70.6
Insafutdinov <i>et al.</i> , CVPR17[20]	88.8	87.0	75.9	64.9	74.2	68.8	60.5	74.3
Cao <i>et al.</i> , CVPR17[7]	91.2	87.6	77.7	66.8	75.4	68.9	61.7	75.6
Newell & Deng, NIPS17[27]	92.1	89.3	78.9	69.8	76.2	71.6	64.7	77.5
ours	88.4	86.5	78.6	70.4	74.4	73.0	65.8	76.7
ours++	91.3	90.5	84.0	76.4	80.3	79.9	72.4	82.1

Table 1. Results on the MPII multi-person test set (mAP). “++” denotes using faster-rcnn with softnms [5] as human detector, PyraNet [45] with input size 320x256 as pose estimator.



課程二：

實作 Human Pose Estimation Model

- Google Colab 平台 實作
- 模型架設/預訓練權重載入
- 熱點圖/換算點座標



GitHub 專案

- **Stacked Hourglass Networks for Human Pose Estimation**

<https://github.com/princeton-vl/pose-hg-train>



- **Pytorch-stacked-hourglass** - 使用PyTorch來執行Stacked Hourglass Networks 的Model

<https://github.com/anibali/pytorch-stacked-hourglass>

- **PyTorch-Pose** - 由PyTorch官方將單人檢測模型做整理，並提供多種資料集的影像、標註數據

<https://github.com/bearpaw/pytorch-pose>





課程大綱 - 人體16個關鍵點計算

Step 1

Import Model / Load package



Step 2

Import package , Function



Step 3

Download weight



Step 4

Define Function



Step 5

Upload Image Image /
preprocessing

Step 6

Output Heatmaps / Estimation
Human Joints



Step 7

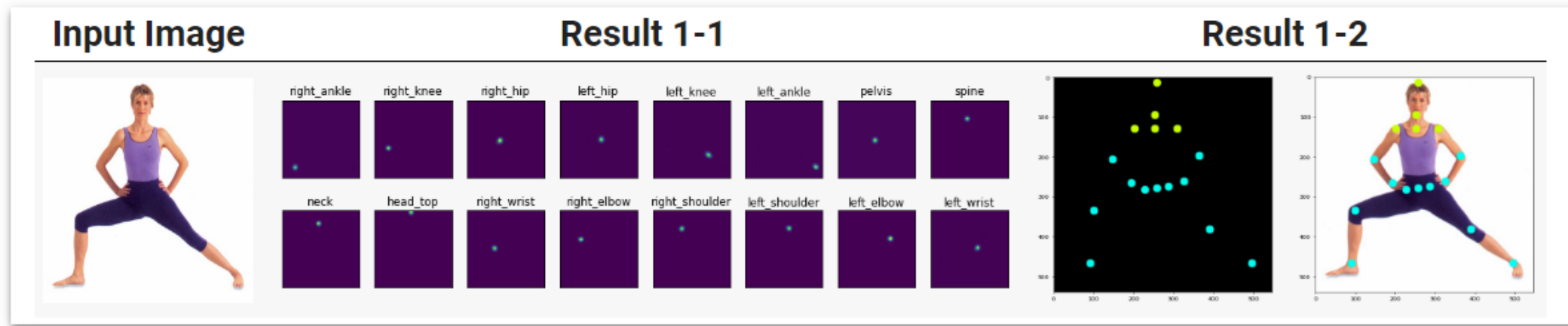
Show Result Image





https://colab.research.google.com/drive/1GTE3UE--ihbE-_p7sp8zSY-b5s4WTC_u

▼ 第二節課 - 實作 Human Pose Estimation Model





使用函數功能說明

函數名稱	主要功能
<code>def calculate_fit_contain_output_area (in_height, in_width, out_height, out_width)</code>	代入原始圖片的長、寬與模型輸出圖的長、寬 計算兩者的比例與偏移值
<code>def estimate_joints (image, heatmaps, flip=False)</code>	代入16個關鍵點的Heatmaps 輸出關鍵點座標值
<code>def draw_human_joints(img, joints)</code>	代入關鍵點座標值標示於原圖上



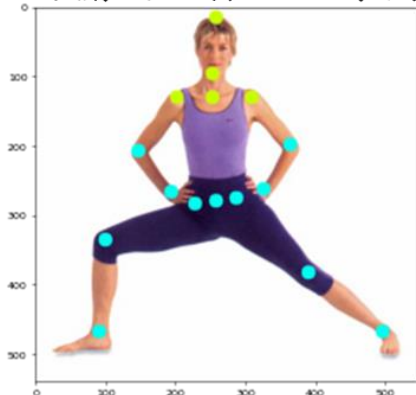
計算16個關鍵點座標

```
heatmaps = predictor.estimate_heatmaps(image_tensor, flip=False)
```

原始圖像



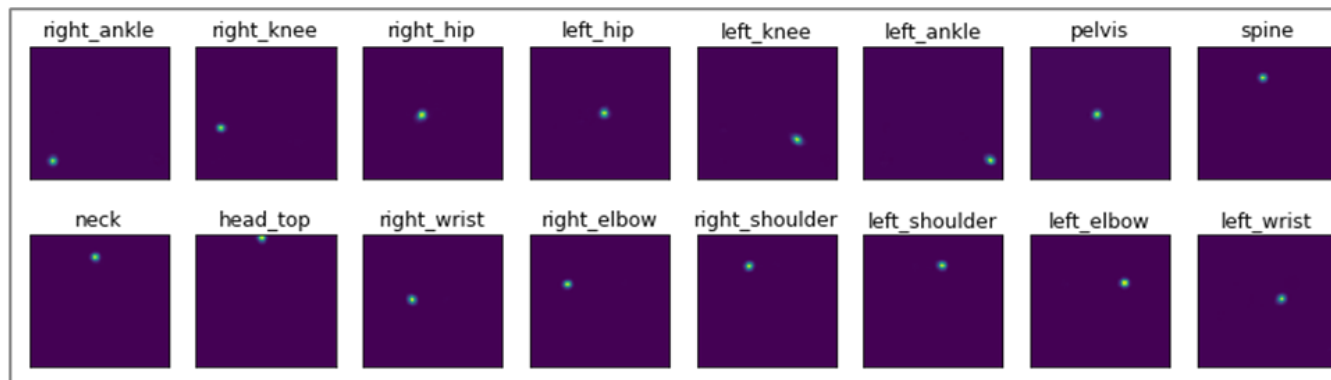
將關鍵點標示於原圖



關鍵點座標

```
right_ankle [91.7109375, 468.4980163574219]  
right_knee [100.2421875, 336.1659851074219]  
right_hip [228.2109375, 284.9407043457031]  
left_hip [287.9296875, 276.40313720703125]  
left_knee [390.3046875, 383.12249755859375]  
left_ankle [496.9453125, 468.4980163574219]  
pelvis [258.0703125, 280.67193603515625]  
spine [253.8046875, 131.2648162841797]  
neck [253.8046875, 97.11461639404297]  
head_top [258.0703125, 16.007904052734375]  
right_wrist [194.0859375, 267.8656005859375]  
right_elbow [147.1640625, 208.10275268554688]  
right_shoulder [202.6171875, 131.2648162841797]  
left_shoulder [309.2578125, 131.2648162841797]  
left_elbow [364.7109375, 199.56520080566406]  
left_wrist [326.3203125, 263.5968322753906]
```

關鍵點熱點圖





課程三：

如何運用 Human Pose Estimation Model

檢測人體偏移

- 利用關鍵點座標計算人體脊椎是否有偏移



課程大綱 - 計算頭部偏移角度

https://colab.research.google.com/drive/1GTE3UE--ihbE-_p7sp8zSY-b5s4WTC_u

Step 1

Define Function



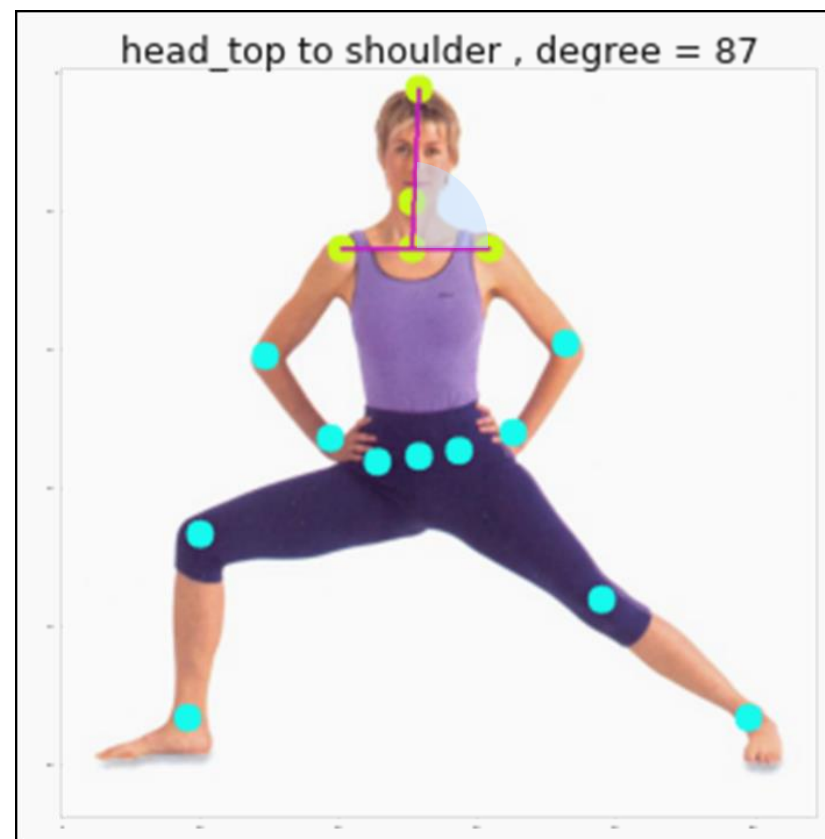
Step 2

讀取關鍵點數據
並計算頭部偏移角度



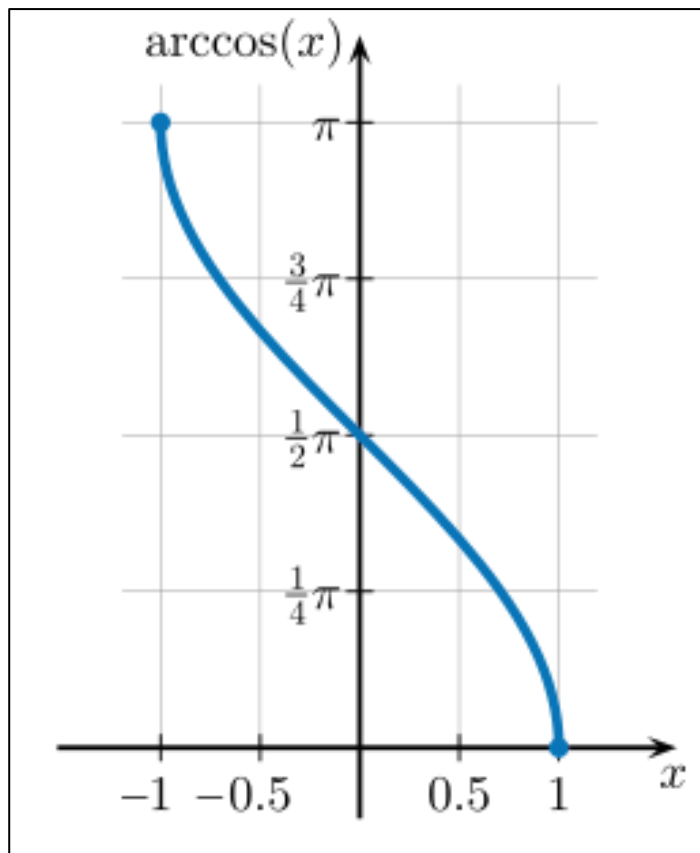
Step 3

將關鍵點連線後顯示結果

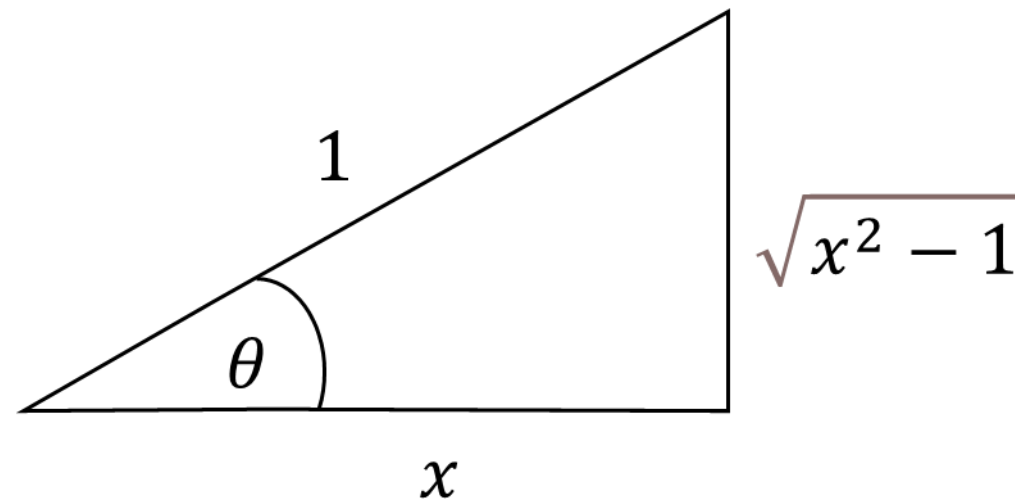




反餘弦計算角度



奇偶性	非奇非偶函數
定義域	$[-1, 1]$
到達域	$[0, \pi]$
週期	N/A



$$\arccos x = -i \ln(x + \sqrt{x^2 - 1})$$



使用函數功能說明

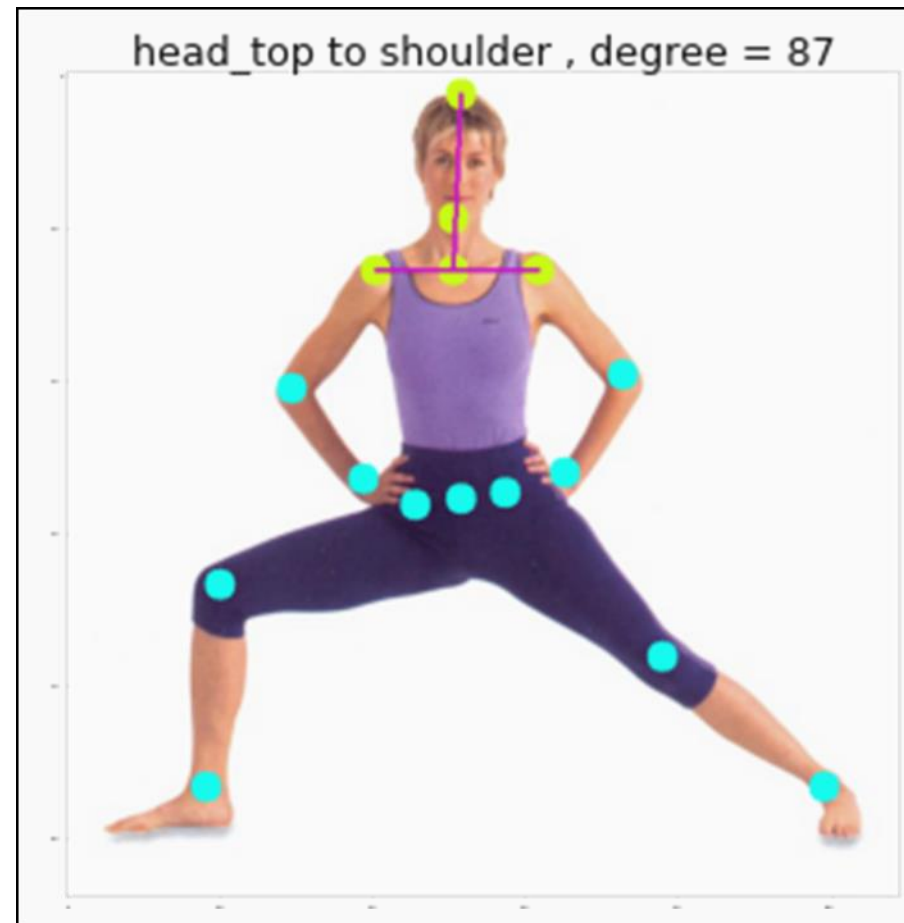
函數名稱	主要功能
<code>def arccos_angle(p1, p2)</code>	代入關鍵點座標，計算兩點向量 並使用反正切函數將向量轉為角度輸出
<code>numpy.arccos(x, /, out=None, *, where=True, casting='same_kind', order='K', dtype=None, subok=True[, signature, extobj])</code>	
<code>def draw_line(img, joints)</code>	代入畫有關鍵點的圖像 將頭部相關的關鍵點連線、輸出角度



計算頭部偏移角度

#計算關鍵點間的角度

```
def arccos_angle(p1, p2):  
    distance = (p1[0] - p2[0])**2 + (p1[1] - p2[1])**2  
    a = np.sqrt([distance])  
    theta = np.sqrt([(p1[0] - p2[0])**2]) / a  
    degree = np.arccos(theta) * 180 / np.pi  
    return degree
```





課程四：

運用 Human Pose Estimation Model 檢測人體側身

- 擴增脊椎檢測點
- 利用關鍵點座標計算人體脊椎健康狀況



課程大綱 - 側身脊椎5點檢測

https://colab.research.google.com/drive/1GTE3UE--ihbE-_p7sp8zSY-b5s4WTC_u

Step 1 Git Clone Project / Install package



Step 2 Import Function



Step 3 Download weight



Step 4 Define Function

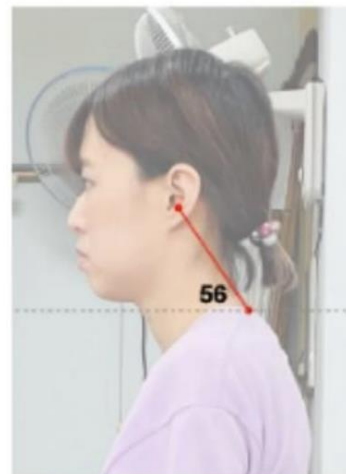


Step 5 Upload Image

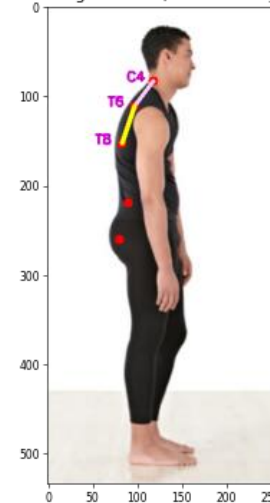
Step 6 Inference



Step 7 Show Result Image



C4 to T6 degree = 53 , T6 to T8 degree = 71





使用函數功能說明

函數名稱	主要功能
<code>def inference(image_path)</code>	代入圖片進行脊椎5點檢測 回傳檢測結果圖與關鍵點座標值
<code>def show_joint_annotation(joints)</code>	代入關鍵點座標值標示於原圖上



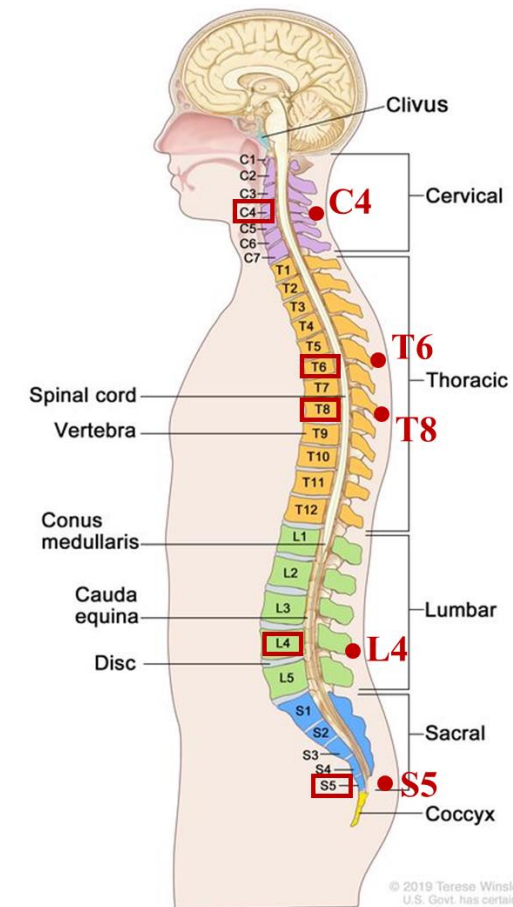
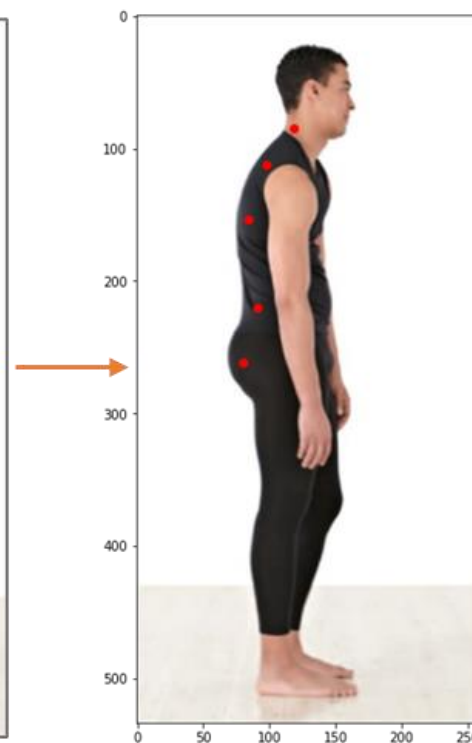
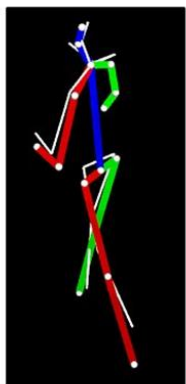
進行脊椎5點檢測

```
!git clone https://github.com/nia630iiii/AIGO_Spine-Detection.git
```

```
from simple_baseline_spine_inference import inference  
  
import os
```

```
image_spine = files.upload()  
img_path = list(image_spine.keys())  
  
inference(img_path[-1])
```

OpenPose



擴增了側面5個關鍵點
[C4,T6,T8,L4,S5]

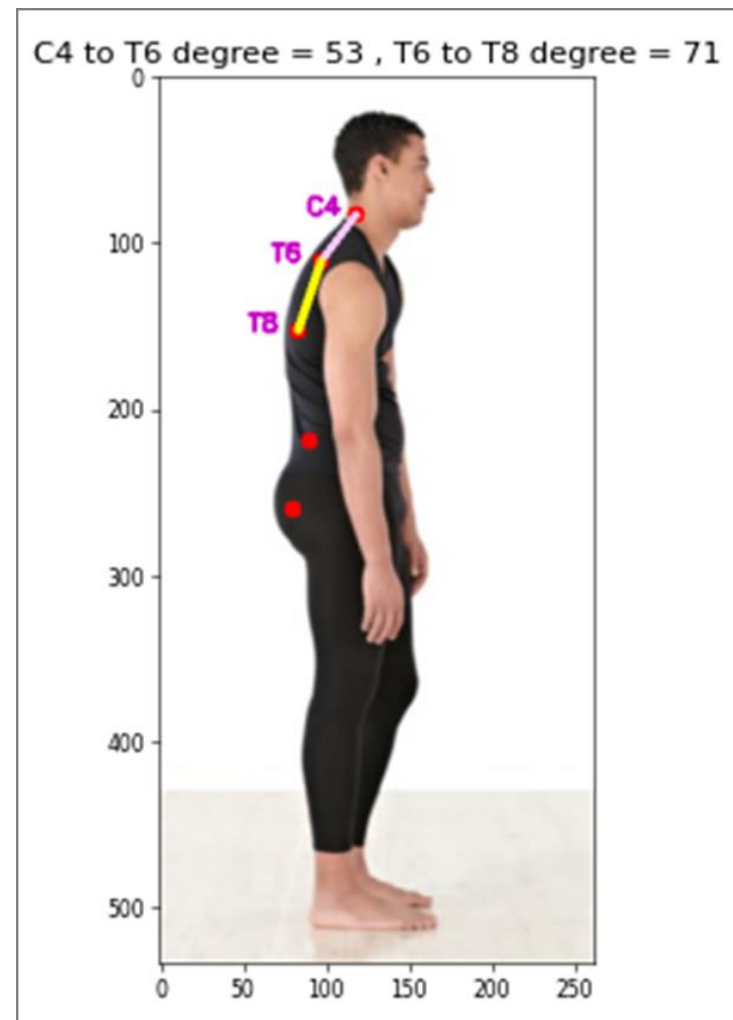


檢測頸椎傾斜角度 (C4,T6,T8)

#計算頸椎傾斜角度

```
def arccos_angle(p1, p2):  
    distance = (p1[0] - p2[0])**2 + (p1[1] - p2[1])**2  
    a = np.sqrt([distance])  
    theta = np.sqrt([(p1[0] - p2[0])**2]) / a  
    degree = np.arccos(theta) * 180 / np.pi  
    return degree
```

```
v1_ang = arccos_angle(spine_joints[:,0],spine_joints[:,1])  
v2_ang = arccos_angle(spine_joints[:,1],spine_joints[:,2])
```



Thank you for your attention

