視訊串流與追蹤 – HW3

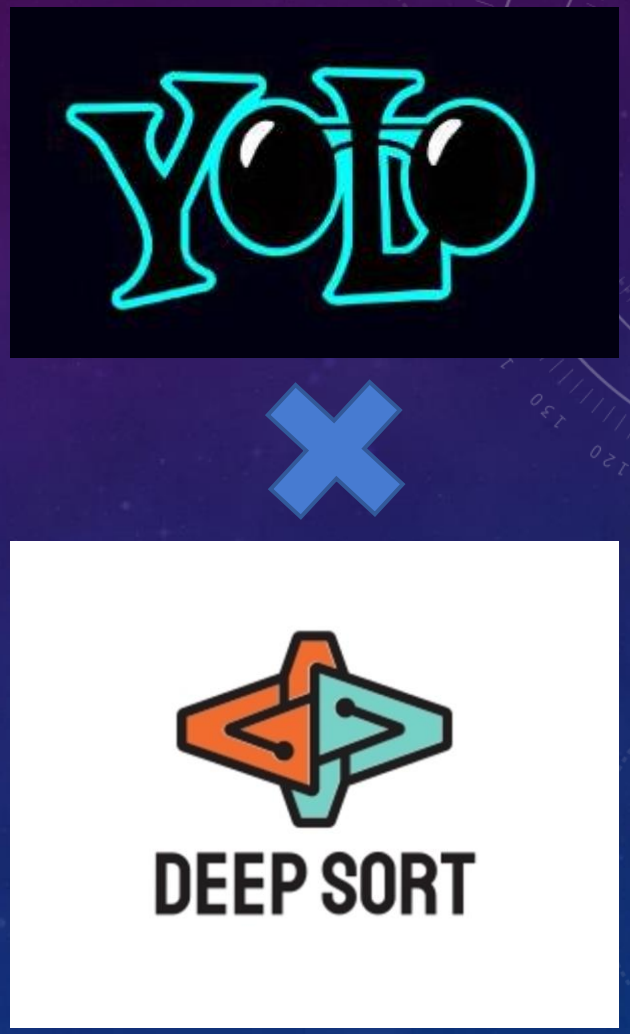
310554004 數據所 許智超

實驗設置EXPERIMENTAL SETUP

模型使用MODEL USED

- Detection部分：**YOLO v4**
- Tracking部分：**Deep Sort**
- 參考程式碼：The AI Guys的github
  - https://github.com/theAIGuysCode/yolov4-deepsort
  - P.S. 繳交的程式碼檔案，只包含我有動手修改的「 **object_tracker_modified.py**」這個檔案。因此若要進行測試，

    還請從github上clone下來，將「**object_tracker.py**」這個檔案替換成我繳交的版本。並照著PPT的後續步驟進行，方得重現！



虛擬環境建置 CREATING ENVIRONMENT

- 為避免套件版本不相容，推薦使用**虛擬環境**
- 可使用conda-cpu.yml或conda-gpu.yml兩個檔案
  - 這裡我用**CPU版本**，因為GPU版本在建置時會發生莫名錯誤.....
- 請先在**Anaconda Prompt**下，使用**cd (路徑)**指令，將工作目錄切換到Clone下來的資料夾
- 接著，照著右圖建置、並激活虛擬環境(擇一即可)

```
# Tensorflow CPU
conda env create -f conda-cpu.yml
conda activate yolov4-cpu

# Tensorflow GPU
conda env create -f conda-gpu.yml
conda activate yolov4-gpu
```

模型權重下載MODEL WEIGHTS DOWNLOAD

- 本source code使用到YOLO v4作物件追蹤
- 請至下列網址下載預訓練好的模型權重檔。
  - https://drive.google.com/open?id=1cewMfusmPjYWbrnuJRuKhPMwRe_b9PaT
- 接著，請放在資料夾「data」下。

模型轉換MODEL CONVERSION

- 在使用模型之前，需要先將YOLO v4中backbone的Darknet部分權重，轉換為Tensorflow版本
  - 因為本source code作者使用Tensorflow操作
- 使用作者寫好的「save_model.py」檔
  - 請見右圖步驟
  - 轉換後的模型，會被放到一個checkpoint資料夾下

- 1. 將前一張slide載好的模型權重，放在資料夾「data」下。
- 2.在Anaconda Prompt下，輸入以下指令

```
# Convert darknet weights to tensorflow model
python save_model.py --model yolov4
```

1. MULTIPLE OBJECT TRACKING ON VIDEOS

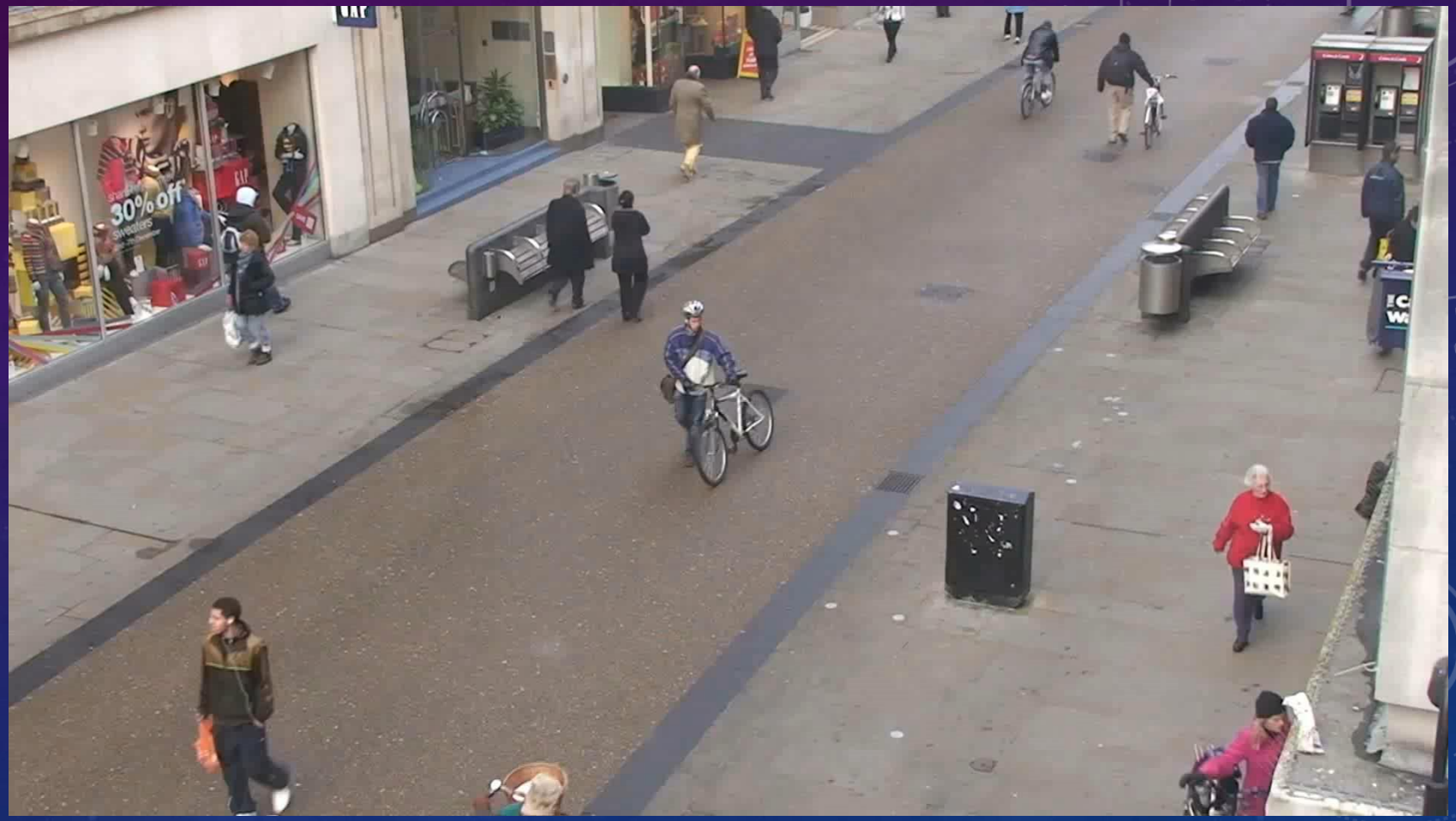
1. 在影片中套用多目標追蹤

MOT ON VIDEOS

- 使用到的是「object_tracker.py」這個檔案
- 請照下圖輸入指令到Anaconda Prompt中
  - Flag 1「--video」：後面放輸入影片的路徑
  - Flag 2「--output」：後面放包含tracking結果的bounding boxes，輸出影片的路徑
  - Flag 3「--model」：指示Detection部分使用的是YOLO v4模型

```
# Run yolov4 deep sort object tracker on video
python object_tracker.py --video ./data/video/test.mp4 --output ./outputs/demo.avi --model yolov4
```

VIDEO DEMO



https://drive.google.com/file/d/1YfG3LH2KiWRQp9ck_1J5_DqN7DJDvaJ-/view?usp=sharing

2. MULTIPLE OBJECT TRACKING ON CAMERAS

2. 在攝像頭上套用多目標追蹤

MOT ON CAMERAS

- 一樣使用到「object_tracker.py」這個檔案
- 請照下圖輸入指令到Anaconda Prompt中
  - Flag 1「--video」：後面放輸入影片的路徑，如果要使用攝像頭，就必須改成0
  - Flag 2「--output」：後面放包含tracking結果的bounding boxes，輸出影片的路徑
  - Flag 3「--model」：指示Detection部分使用的是YOLO v4模型

```
# Run yolov4 deep sort object tracker on webcam (set video flag to 0)
python object_tracker.py --video 0 --output ./outputs/webcam.avi --model yolov4
```

VIDEO DEMO

https://drive.google.com/file/d/1JHxVuM44D-vSMuGf_gKUbI1Fxc63gaLe/view?usp=sharing

3. TRACK ONE OBJECT CHOSEN BY USER
4. CANCEL THE OBJECT SELECTED BY USER

3. 點擊滑鼠左鍵、追蹤使用者選取的單一目標

4. 再點一次滑鼠左鍵、取消選取該目標、繼續追蹤多目標

這2部分，請使用我修改後的CODE.PY

就是我上傳的OBJECT_TRACKER_MODIFIED.PY

用來替換掉「OBJECT_TRACKER.PY」這個檔案

新增部分的更動想法：

- 其實我並沒有真的只追蹤1個目標，而是一樣追蹤所有目標，但**只顯示被選中那個**
  - **也可達到一樣的視覺效果！**
- 1. 用一個Boolean變數，決定是否只顯示單目標
- 2. 偵測並記錄滑鼠點擊的位置 (**opencv套件中的函數**)
- 3. 紀錄把點擊位置框住的那個bounding box的tracker id
- 4. 只顯示被選中的bounding box，其他的一樣偵測，但跳過不顯示！
  - 以下數張slides將詳解更動部分的程式碼

更動部分程式碼詳述：變數定義

- 首先，在main function外面，新增3個**全域變數(global variables)**
  - single: 紀錄是否只顯示單一物件
    - 預設值為**False**
  - point: 紀錄本次滑鼠點擊的位置
    - 點擊後將會有x, y兩個值
    - 預設值為**空串列[]**
  - trackerid: 紀錄本次滑鼠點擊的位置對應到哪個目標，他的id。
    - 預設值為**-1**

```
single = False
point = []
trackerid = -1
def main(_argv):
```

更動部分程式碼詳述：滑鼠點擊的函數

- 定義函數**mouseClicked()**，參數有
  - **event**: 紀錄所有滑鼠事件的代號
  - **x, y**: 滑鼠位置的座標值
- 首先，如果我們偵測到**滑鼠左鍵點擊**
- 就把**全域變數**single, point, trackerid叫進來
- 如果現在single = False, 也就是目前是全部追蹤
  - 就把single改成True, 也就是只追蹤單一目標
  - 並且把滑鼠點擊的x, y座標放入point

```
def mouseClicked(event, x, y, flags, params):
    if event == cv2.EVENT_LBUTTONDOWN:
        global single, point, trackerid
        if not single:
            single = True
            point.append(x)
            point.append(y)
        else:
            single = False
            point = []
            trackerid = -1

frame_num = 0
# while video is running
while True:
    cv2.setMouseCallback("Output Video", mouseClicked)

    return_value, frame = vid.read()
    if return_value:
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
```

更動部分程式碼詳述：滑鼠點擊的函數

- 如果點擊當下的single = True, 表示目前是單一目標追蹤，但要改回全部追蹤
  - 故把single改回False, 也就是全部追蹤
  - 並且把point清空、把trackerid改回-1
- 接著，在處理影片每幀的while迴圈中，放入opencv的函數**setMouseCallback**
  - 用來**捕捉滑鼠相關的事件**
  - 把剛剛定義的函數mouseClicked丟進去

```
def mouseClicked(event, x, y, flags, params):
    if event == cv2.EVENT_LBUTTONDOWN:
        global single, point, trackerid
        if not single:
            single = True
            point.append(x)
            point.append(y)
        else:
            single = False
            point = []
            trackerid = -1

frame_num = 0
# while video is running
while True:
    cv2.setMouseCallback("Output Video", mouseClicked)

    return_value, frame = vid.read()
    if return_value:
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
```

更動部分程式碼詳述：追蹤選取的目標

- **紅線部分以下**是我新增的程式碼區塊
- 對於每個追蹤中的物體：
  - 如果single = True，代表滑鼠被點擊了(因為預設是False)，那麼就**只需要顯示被選取的**單一目標的追蹤。
  - 把**全域變數**trackerid拿進來，等下要判斷+處理

```
# update tracks
for track in tracker.tracks:
    if not track.is_confirmed() or track.time_since_update > 1:
        continue
    bbox = track.to_tlbr()
    class_name = track.get_class()
    # if single:
    #     if int(bbox[0]) > point[0] or point[0] > int(bbox[2]) or int(bbox[1]) > point[1] or point[1] > int(bbox[3]):
    #         continue
    if single:
        global trackerid
        if trackerid == -1 and int(bbox[0]) <= point[0] and point[0] <= int(bbox[2]) and int(bbox[1]) <= point[1] and point[1] <= int(bbox[3]):
            trackerid = track.track_id
        if track.track_id != trackerid:
            continue
```

更動部分程式碼詳述：追蹤選取的目標

- 如果trackerid是-1(表示還沒被動過)、以及滑鼠點擊的位置座標(point)落在當前的bounding box，表示這個bounding box對應到物體是被選取的，那就
  - 將trackerid的值改成現在這個bounding box的id。
  - 如果偵測到的bounding box不是這個id，就用continue跳過，不進行之後畫出bounding box的動作

```
# update tracks
for track in tracker.tracks:
    if not track.is_confirmed() or track.time_since_update > 1:
        continue
    bbox = track.to_tlbr()
    class_name = track.get_class()
    # if single:
    #     if int(bbox[0]) > point[0] or point[0] > int(bbox[2]) or int(bbox[1]) > point[1] or point[1] > int(bbox[3]):
    #         continue
    if single:
        global trackerid
        if trackerid == -1 and int(bbox[0]) <= point[0] and point[0] <= int(bbox[2]) and int(bbox[1]) <= point[1] and point[1] <= int(bbox[3]):
            trackerid = track.track_id
        if track.track_id != trackerid:
            continue
```

CLICK TO SELECT, ONE MORE CLICK TO CANCEL

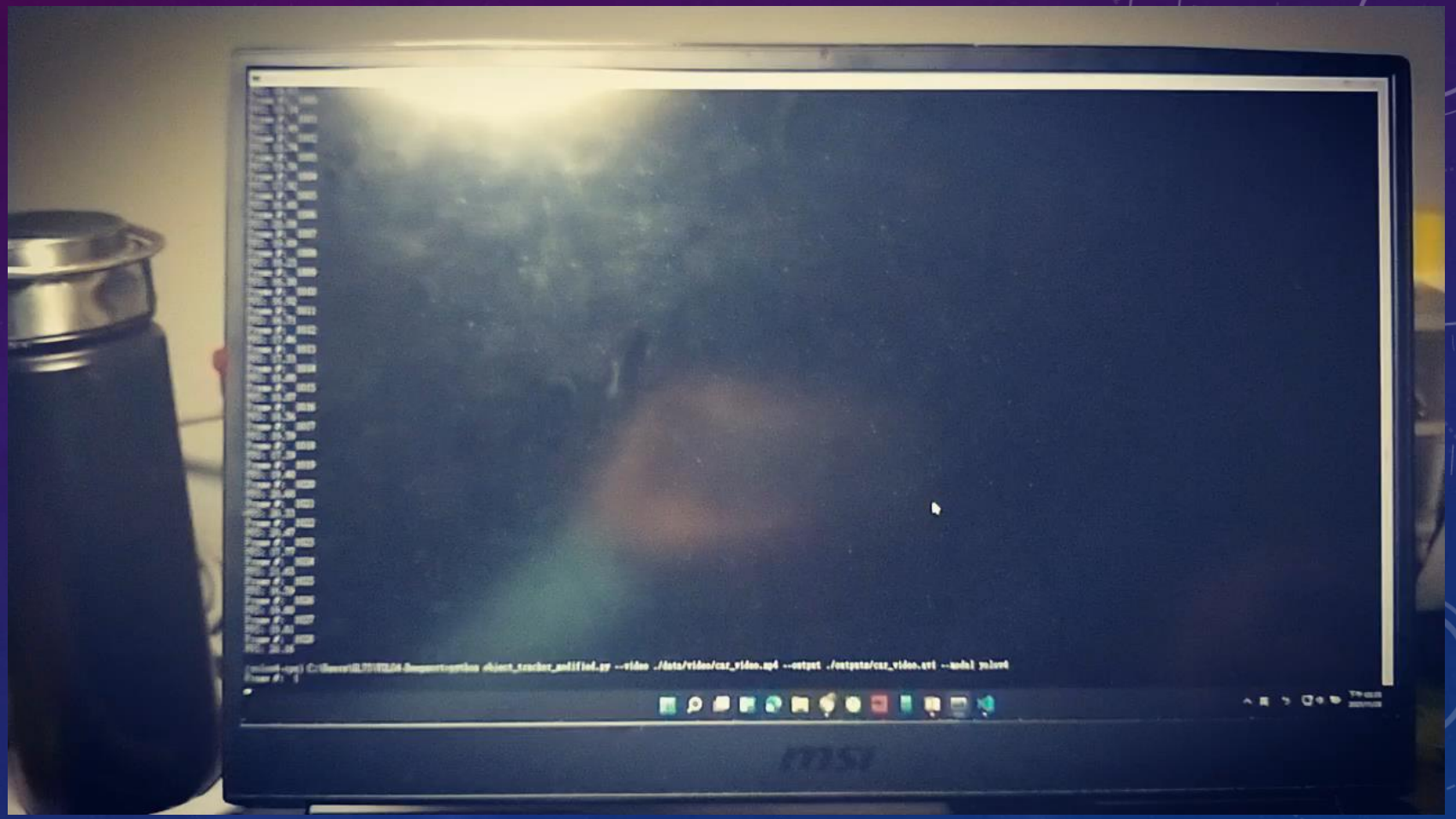
- 請使用到我修改過的「 object_tracker_modified.py」這個檔案！
- 請照下圖輸入指令到Anaconda Prompt中
  - Flag 1「 --video」：後面放輸入影片的路徑 ，請改成自己的影片！(如果要使用**攝像頭**，請改成**0**)
  - Flag 2「 --output」：後面放包含tracking結果的bounding boxes，**輸出影片**的路徑
  - Flag 3「 --model」：指示Detection部分使用的是**YOLO v4**模型

```
python object_tracker_modified.py --video ./data/video/car_video.mp4 --output ./outputs/car_video.avi --model yolov4
```

  - 註：我是用偵測車子的影片，想要偵測其他東西，可到第181行更改
    此外，如果要偵測所有東西，可將第181行註解掉，並將第178行取消註解。
  - 按Q中斷影片追蹤

VIDEO DEMO

你可以聽到滑鼠點擊聲
以及看到我滑鼠的位置



https://drive.google.com/file/d/1_44s9OVMPBhJ19HYpUol1gNnYDwLR9uU/view?usp=sharing

PROBLEM AND DISCUSSION

問題與討論

- 由於大多數是借助The AI Guys的Github程式碼，沒什麼大問題。
- 但為了增加**選取/取消追蹤單一目標**的功能，我還是有修改了一些部份，本來的想法是紅色框框

```
# update tracks
for track in tracker.tracks:
    if not track.is_confirmed() or track.time_since_update > 1:
        continue
    bbox = track.to_tlbr()
    class_name = track.get_class()
    # if single:
    #     if int(bbox[0]) > point[0] or point[0] > int(bbox[2]) or int(bbox[1]) > point[1] or point[1] > int(bbox[3]):
    #         continue
    if single:
        global trackerid
        if trackerid == -1 and int(bbox[0]) <= point[0] and point[0] <= int(bbox[2]) and int(bbox[1]) <= point[1] and point[1] <= int(bbox[3]):
            trackerid = track.track_id
        if track.track_id != trackerid:
            continue
```

- 像這樣，直接判斷滑鼠按下的點，是否落在某個bounding boxes的範圍就好，但

- **實測**之後，發現這會產生一個小bug：
  - 因為**滑鼠點下的點**是**固定的**，但是選取的**目標**、以及對應的**bounding box**會**隨著幀數移動**！
- 這樣會造成在未來的某一幀，該bounding box極可能**移動到不再包含滑鼠點的位置**
- 於是，照著程式的邏輯，
  - 1. 這個**選取的**bounding box，**就此消失不被畫出來！**
  - 2. 其他**不是我們點選的**bounding box也會移動，可能**移動到包含滑鼠點的位置**，結果**反而被顯示出來**！
- 總之，就是會發生選取的目標不畫出來、反而去畫到其他目標的追蹤結果

- **解決方法**：點擊當下**還是要使用滑鼠位置**來確定是哪個bounding box，但是多紀錄它的**id**
  後續的單一追蹤，**就靠追蹤這個id去達成**，而不是靠著滑鼠點下的位置有哪些bounding boxes經過！

- 另外一個小問題，就是我在使用攝像機的時候，發現他誤將我的手機也偵測成「人」
- 但這可能跟模型本身有關，因此這個問題就比較無解了.....