**Color Doppler API**

**包含模組:**

* **integration\_v1.py** (主要演算法的檔案)
* **model\_main.py** (使用 integration\_v1.py 的主要檔案)
* handle\_degree.py (給醫生展示時的結果測試)

(doppler\_curr2.py & integration\_main.py 皆為開發階段測試使用)

**簡介:**

為了診斷心臟中的腔室血液是否逆流以及嚴重程度，找出影像中彩色都卜勒區域，對顏色的值進行分析來判斷逆流的深度及面積。

**使用方式:**

模組名稱: integration\_v1.py

此模組使用Python Class來創建，Class 名稱為DopplerModel、DicomData，以下為兩者的必要輸入參數:

* **Class Name: DopplerModel**

|  |  |  |
| --- | --- | --- |
| 變數名稱 | 資料類型 | 說明 |
| video\_path | str | 輸入影片路徑 |
| diagnosisPos | str | 輸入診斷位置(在還沒有 Segmentation 時，必須靠手動分類區分後才能使用) |
| case\_name | str | 輸入病歷號碼 |

Example:



doppler: 變數名

path: 影片路徑

curr\_case\_name: 當前病歷號碼(每個病例皆屬於一個資料夾，該變數為資料夾名稱)，如圖1

curr\_pos: 當前診斷位置(手動分類完，並創建每個診斷位置的資料夾，該變數為資料夾名稱)，如圖2

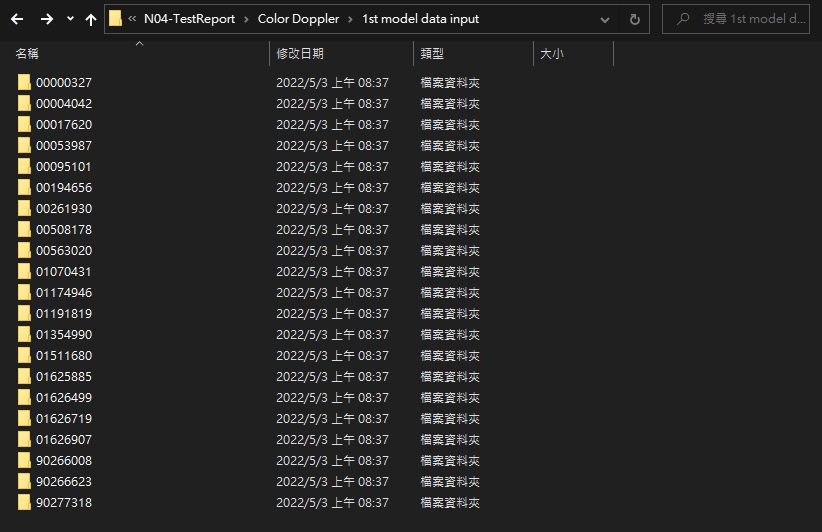


圖1. 病歷資料夾名稱



圖2. 診斷位置的資料夾名稱

Aortic: 動脈

Mitral: 二尖瓣

Pulmonary: 肺動脈

Tricuspid: 三尖瓣

(實際上是指瓣膜位置，省去Valve部分)

* **Method**:

1. **get\_frame(self): 讀取每幀影像**

**return:** 當前 frame，numpy.ndarray

Example:



此做法只是省略掉原本使用 cv2 模組時，需要用 ret, frame = video.read() 的前面 ret 變數，且把 ret 變成屬性的一部份。(可以使用個人習慣做法)

1. **find\_region(self): 找出 Color Doppler 有效扇形區域**

**return**: Color Doppler 有效區域 mask，numpy.ndarray，二值圖

Example:



mask\_region: 變數名

(使用物件呼叫方法即可)

1. **standard\_unit(self, unit\_len): 調整影像標準長度(pixel 轉成 cm)**

**parameter:**

unit\_len: 當前影像的標準單位長度 cm，int (以手動輸入方式處理)

**return:** None

Example:



unit\_list[case\_index][curr\_index][i]: 手動方式輸入時，透過列表來事先打好所有影像的標準長度，用迭代查表。(可以用個人習慣做法)

(使用物件呼叫方法即可)

1. **color\_info(self, frame[, HueRange=(26, 99)[, maxDis=60]]):**

**找出有問題的顏色區域**

**parameters:**

frame: 當前影像(經過遮罩後)

HueRange: HSV的色相(H)範圍。紅(H=0)、藍(H=124)，默認採用(26, 99) 區間，tuple(int, int)。

maxDis: 有問題區域中的輪廓距離門檻值，默認60，int

**return**:

mask\_connect: 傳回 connect 後的遮罩，numpy.ndarray，二值圖

Example:



frame[mask\_region != 255] = [0, 0, 0]

frame 為原始影像，採用 numpy 的 mask 做法對 frame 做遮罩，也可以使用cv2的 cv2.bitwise\_and(frame, mask\_region)，但兩者的通道數必須相同才不會出錯。由於mask\_region 是灰階，要轉成BGR三通道才可以用cv2的做法。

mask\_connect = doppler.color\_info(frame)

mask\_connect: 變數名

使用物件呼叫方法，參數僅給經遮罩後的圖像，其餘不調整。

1. **show(self, frame, mask\_connect[, minArea=0.1[,personalInfo=False,[dcm\_path=None]]]): 展示 Doppler 結果**

**parameters:**

frame: 每幀三通道原始影像

mask\_connect: 使用obj.color\_info() 的回傳結果

minArea: 有問題區域的最小面積，默認為 0.1cm2，float

personalInfo: 是否顯示個人資訊，默認False，bool

dcm\_path: DCM檔案路徑，默認 None，str

**return:**

curr\_res: 當前呈現結果，numpy.ndarray

Example:



result: 變數名

original: 原始三通道影像

使用物件呼叫方法，並且傳入相應參數即可使用。

1. **show\_TimeBar(self, result\_list): 顯示具體的展示資訊**

**parameter:**

result\_list: 已經處理過的結果列表，list(numpy.ndarray)

**return:**

curr\_res: 當前的展示結果，list(numpy.ndarray)

Example:



將使用show() 方法得到的輸出，使用Python list 儲存後，傳入給result\_list 參數。



1. **write\_video(self, frames, output\_path[, fps=30]): 寫入影片到指定路徑**

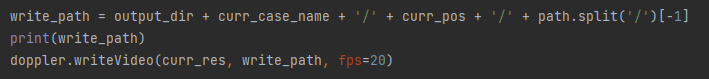
**parameters:**

frames: 儲存所有frame的列表，list(numpy.ndarray)

output\_path: 影片輸出路徑(**不能使用階層資料夾**，這部分在這裡並未實作)，str

fps: frames per second，每秒幾個畫面，默認30，int。fps越小影片速度越慢。

Example:



1. **gen\_txt(self, txt\_path): 產生txt報告(當前已沒再使用)**
2. **degree(self): 診斷嚴重程度，此method已在show() 裡面被呼叫，因此不需要再額外呼叫一次**