視覺

目的是得到物件的座標點，物件的座標點用途
(1) 偵測箱子是否被推歪
(2) 搭配上 Zenbo 的人臉偵測(可得知相對於 Zenbo 的 x, y, z 值)，如果玩家不會走動，而遊戲一開始 Zenbo 偵測到玩家的臉、記錄下 x,y,z 值，便可能做出遊戲過程中 Zenbo 時不時轉頭看看玩家、作做不同表情的效果
(3) 防呆：得知各物件位於哪一個格子，來確認是否與程式另外紀錄的倉庫狀態矩陣吻合，避免一開始將地圖和物件擺錯位置

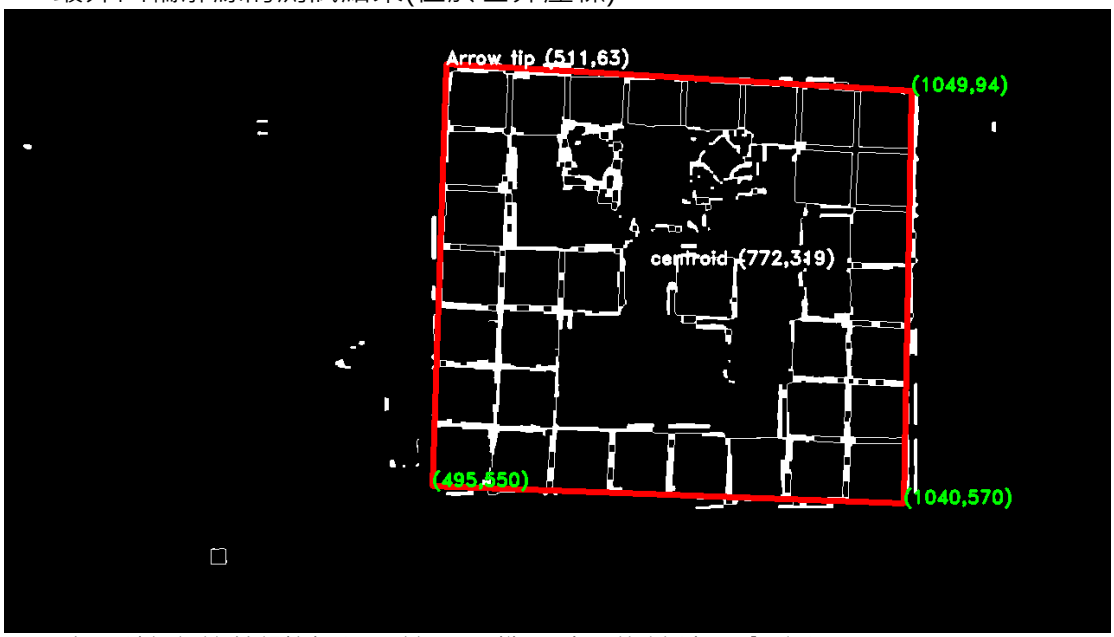
1_preparation.py

- 用途：架設相機後校正畫面
- 場景條件：

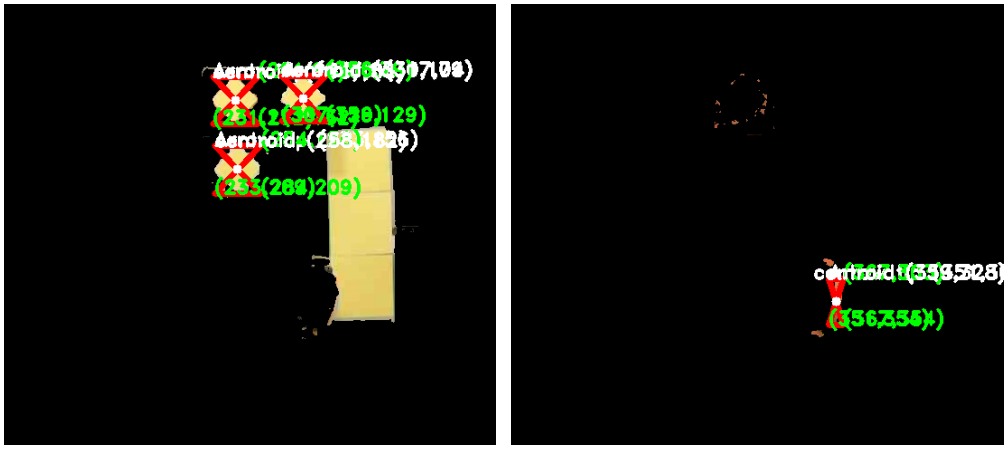
(1) 畫面中無強烈反光，如此程式判斷物件顏色才比較準確
(2) 需要 chessboard 做內部外部參數校正
(3) 需要倉庫的紙本地圖做透視視角的修正(轉正、以及將梯形修復回方形)，程式預設偵測倉庫的圍牆(綠色)

- 測試圖檔：Vision/camera_data 資料夾下
  monitoring_0.png, monitoring_1.png, monitoring_2.png
- 測試

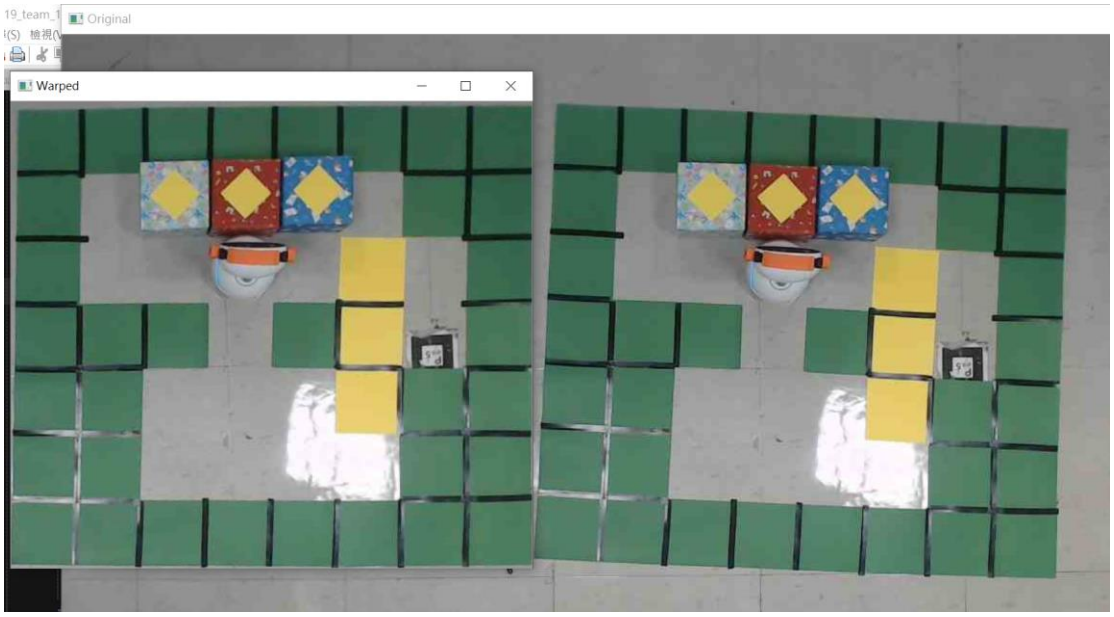
(1) 物件頂點和中心點的偵測，畫面左上角為座標點(0,0)
最外圈輪廓線的測試結果(位於世界座標)

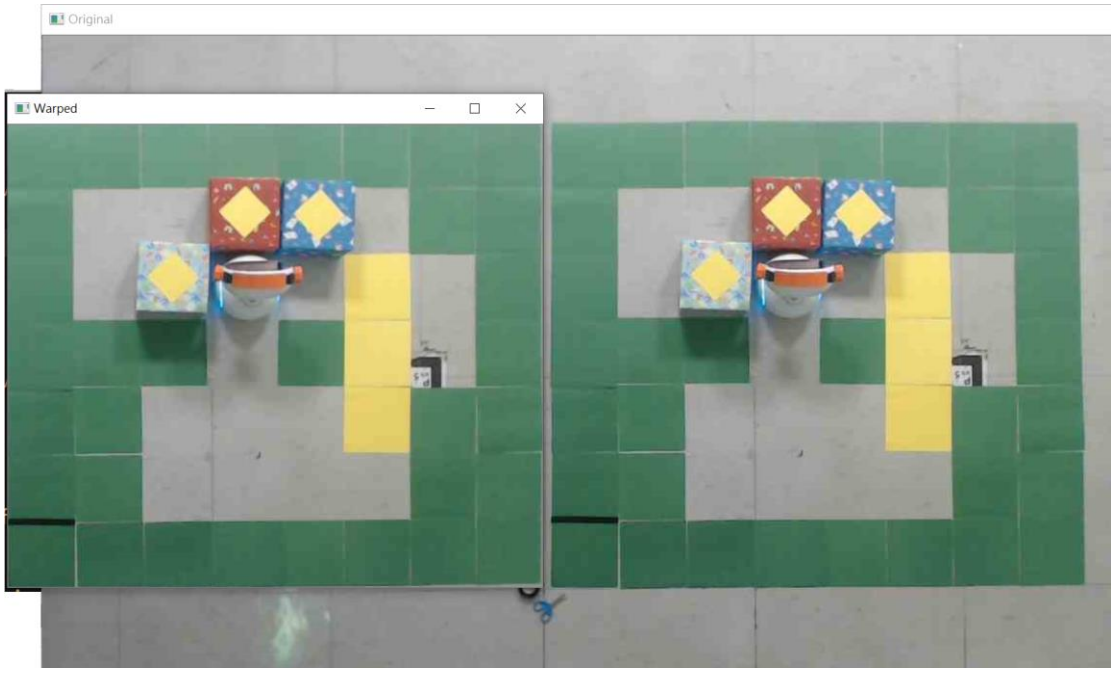


個別輪廓線的測試結果(箱子、機器人)(位於地圖內座標)



(2) 透視視角的修正效果：





(3) 判斷物件在地圖中的哪一個格子(以格子的 index 表示，畫面左上角為(0,0))

測試檔	monitoring_1	monitoring_2
Zenbo	(3,2)	(4,4)
箱子	(4,4)(3,5)(4,5)	(4,5)(3,5)(5,4)

- 已知問題：

(1) 顏色：程式不能乾淨地分辨箱子的紅色和機器人的橘色，不論開燈或是關燈皆如此，增加用物件的大小來判斷是紅色箱子或橘色機器人；程式不能分辨水藍色和機器人、地板的白色

(2) 物件頂點的誤差來自於
   i. 透過顏色選取物件的效果
   ii. cv2.approxPolyDP 函式的 epsilon 設定值→用二元搜尋法來最佳化 epsilon 值?

從小畫家用人眼判斷圍牆的頂點座標	程式結果	誤差像素
518, 71	511, 63	-7, -8
1053, 98	1049, 94	-4, -4
494, 546	495, 550	+1, +4
1040, 567	1040, 570	0, +3

(3) 箱子被推歪之後 Zenbo 才修正身體面向，這樣會太遲，Zenbo 沒有手，很難將箱子喬正 → 假設箱子一開始是擺正的，在推箱子之前，Zenbo 得將自己的身體轉正