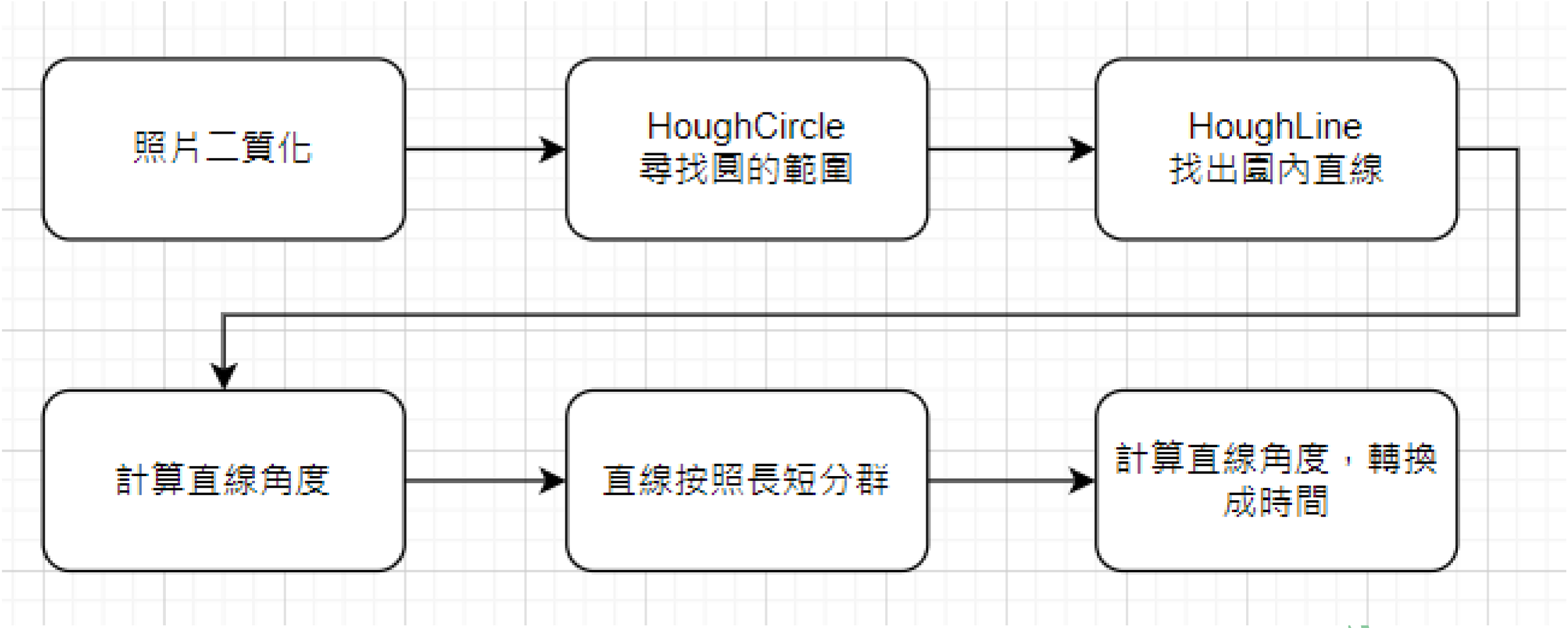
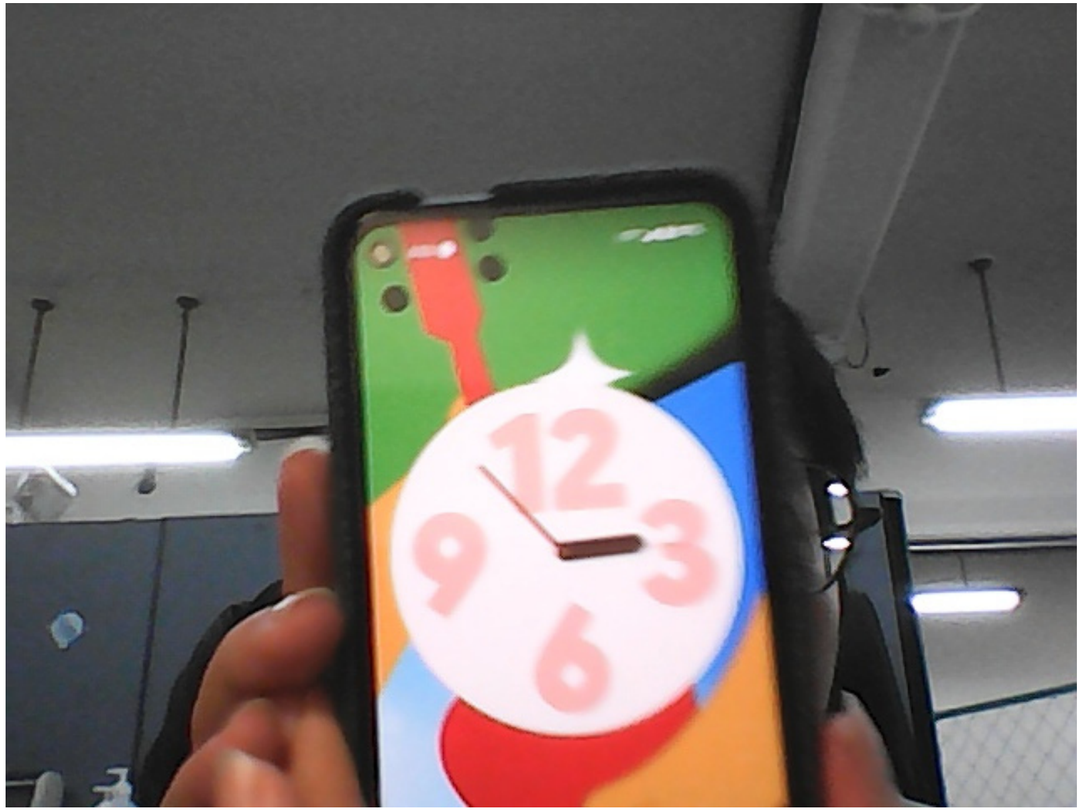
電腦視覺辨識時間

方法一步驟


照片二質化
HoughCircle
尋找圓的範圍
HoughLine
找出圓內直線
計算直線角度
直線按照長短分群
計算直線角度，轉換
成時間

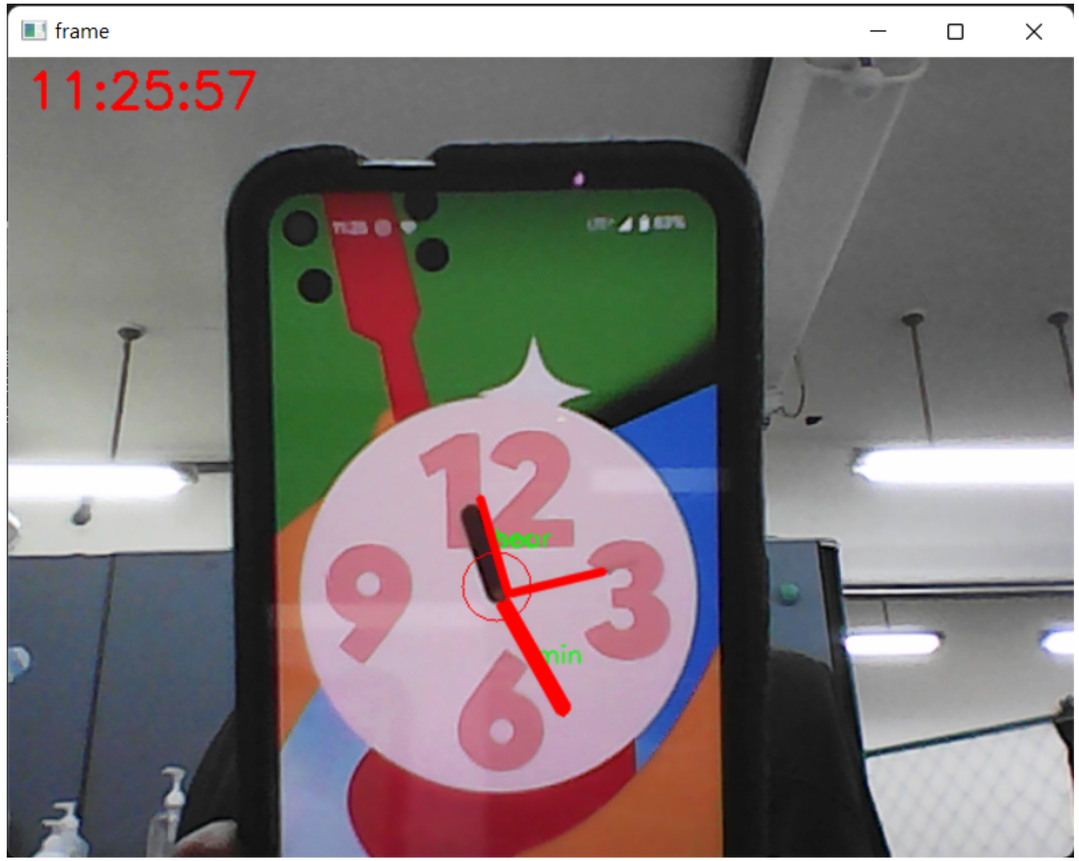
輸入圖片

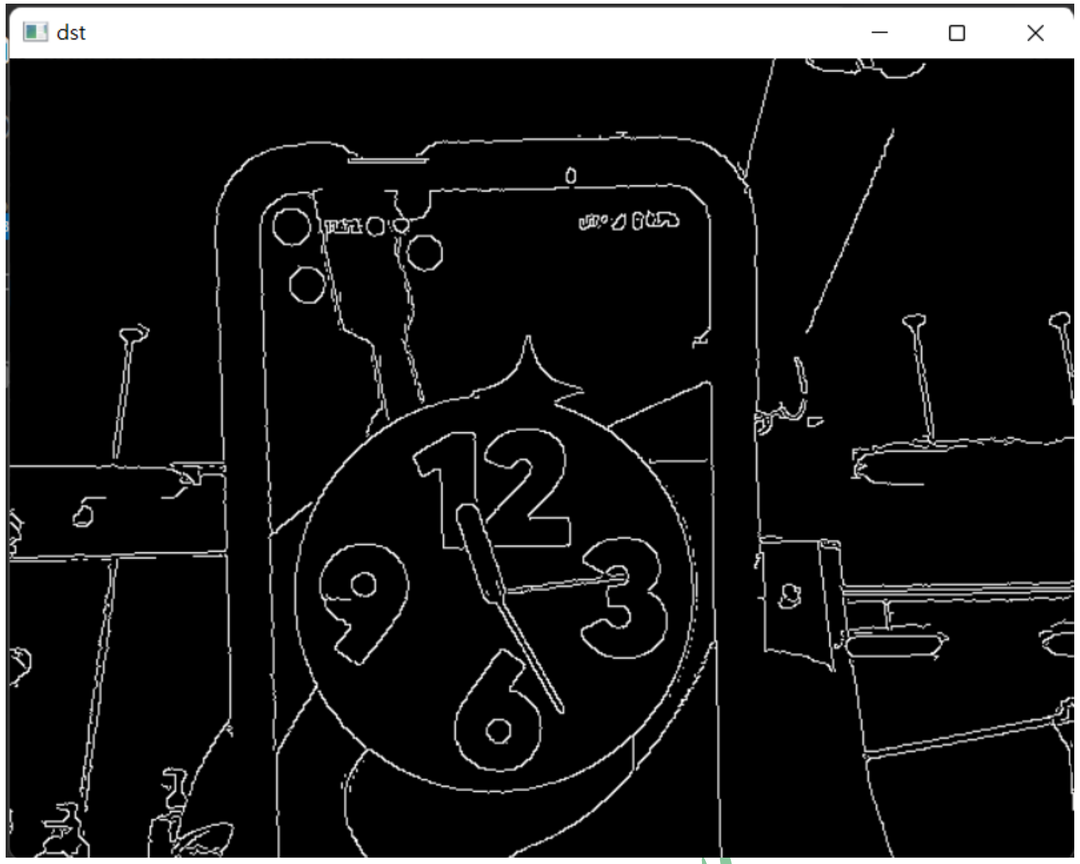
使用相機捕捉畫面



影像處理

cv2.canny 邊緣檢測





影像處理

尋找圓圈

設定範圍在80~200pixels

```python
def findCircle(img):
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    minDist = 100
    param1 = 40  # 500
    param2 = 50  # 200 #smaller value-> more false circles
    minRadius = 80
    maxRadius = 200  # 10      You, 前天 • update

    circles = cv2.HoughCircles(gray, cv2.HOUGH_GRADIENT, 1, minDist,
                               param1=param1, param2=param2, minRadius=minRadius, maxRadius=maxRadius)
    return circles
```

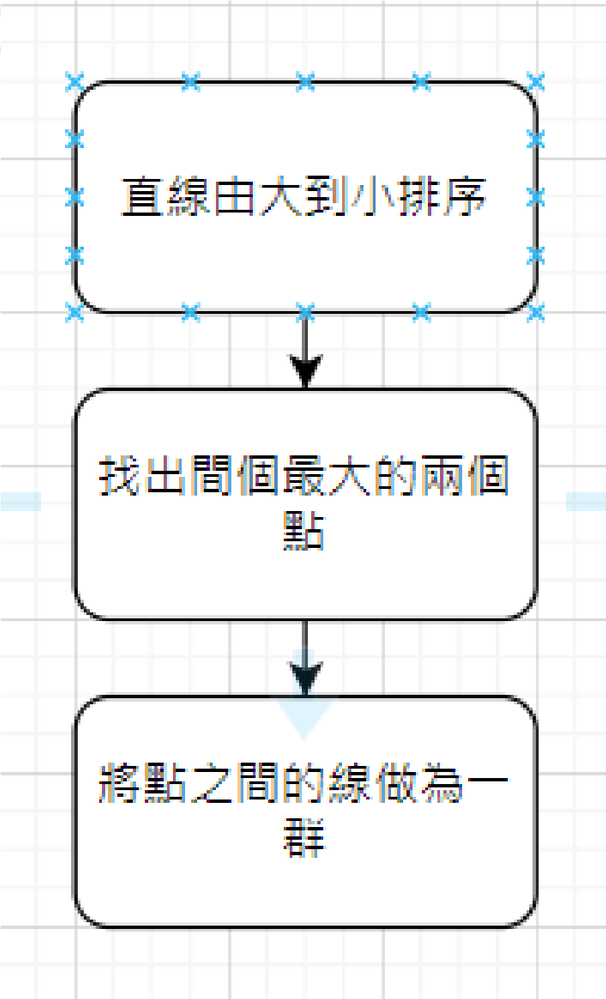
尋找直線

設定範圍在50pixels以上

```python
def findLines(img):
    dst = cv2.Canny(img, 50, 200, None, 3)
    cv2.imshow('dst', dst)
    cv2.waitKey(0)
    # Copy edges to the images that will display the results in BGR
    cdst = cv2.cvtColor(dst, cv2.COLOR_GRAY2BGR)
    cdstP = np.copy(cdst)

    lines = cv2.HoughLines(dst, 1, np.pi / 180, 50, None, 0, 0)
```

直線分群




```
# ---------------------------------- 將每條線從大到小排序 ---------------------------------- #
for line in lines:
    length = math.sqrt((line[0]-line[2])**2+(line[1]-line[3])**2)
    lines_len.append(length)
    if len(lines_sort) == 0:
        lines_sort.append((length, line))
        continue
    for lenidx in range(len(lines_sort)):
        if length > lines_sort[lenidx][0]:
            lines_sort.insert(lenidx, (length, line))
            break
        if lenidx == len(lines_sort)-1:
            lines_sort.append((length, line))
# ---------------------------------- 長度分群 ---------------------------------- #
dividing_idx = [0,0]  # 分群點 [座標]
difs=[0,0]  # 分群點 [差距]
for i in range(len(lines_sort)-1):
    dif = lines_sort[i][0] - lines_sort[i+1][0]
    if dif > difs[1]:
        dividing_idx[1] = i
        difs[1] = dif
    elif dif > difs[0]:
        dividing_idx[0] = i
        difs[0] = dif
print('dividing_idx: ',dividing_idx)
for line in lines_sort:
    print('line: ',line)
#                                                                                    #
```

角度計算時間

計算線的tan

```
sec_tan = -(sec_line[1][3] - sec_line[1][1]) /
(sec_line[1][2] - sec_line[1][0])
```

將tan轉換成時鐘對應的弧度

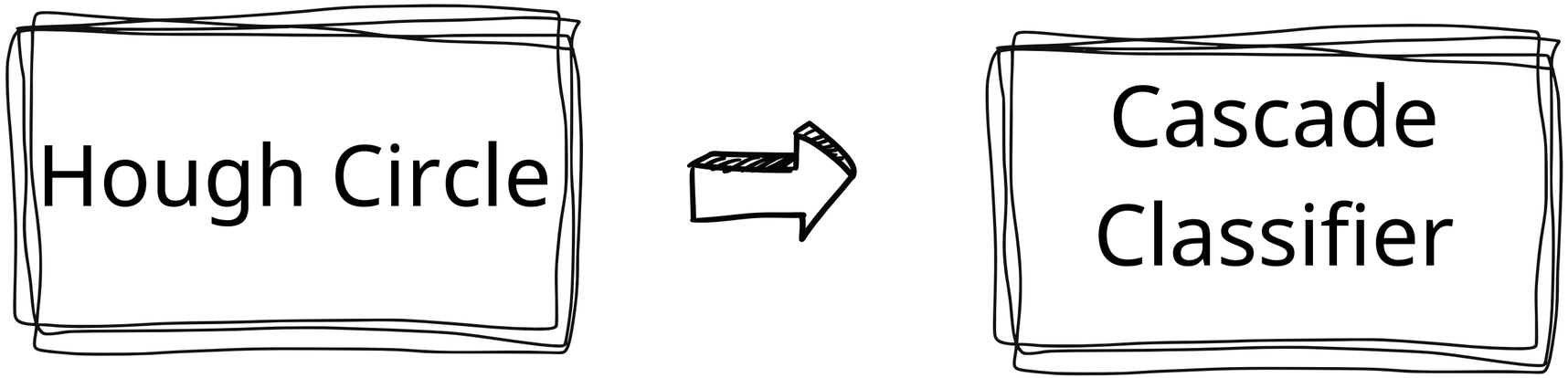
```
sec_angle = 2*math.pi -
math.atan(sec_tan) + math.pi/2
```

一圈60秒

```
sec_time = int(sec_angle/(2*math.pi/60)) % 60
```

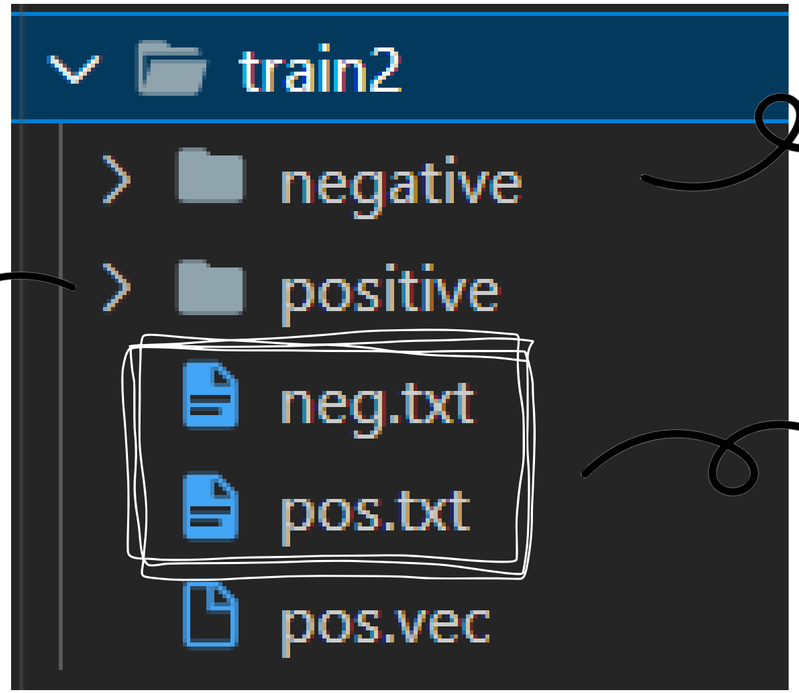
方法二

尋找圓圈改爲訓練模型尋找時鐘



模型訓練

資料集



無時鐘的照片集

有時鐘的
照片集

照片路徑

模型訓練

框物件

1. opencv_annotation --annotations=pos2.txt --images=train2/positive/

增加訓練樣本

2. opencv_createsamples -info pos.txt -w 24 -h 24 -num 1000 -vec pos.vec

訓練模型

3. opencv_traincascade -data ../cascade/ -vec pos.vec -bg neg.txt -w 24 -h 24 -numPos 50 -numNeg 50 numStages 10

結果