



# Clock recognition

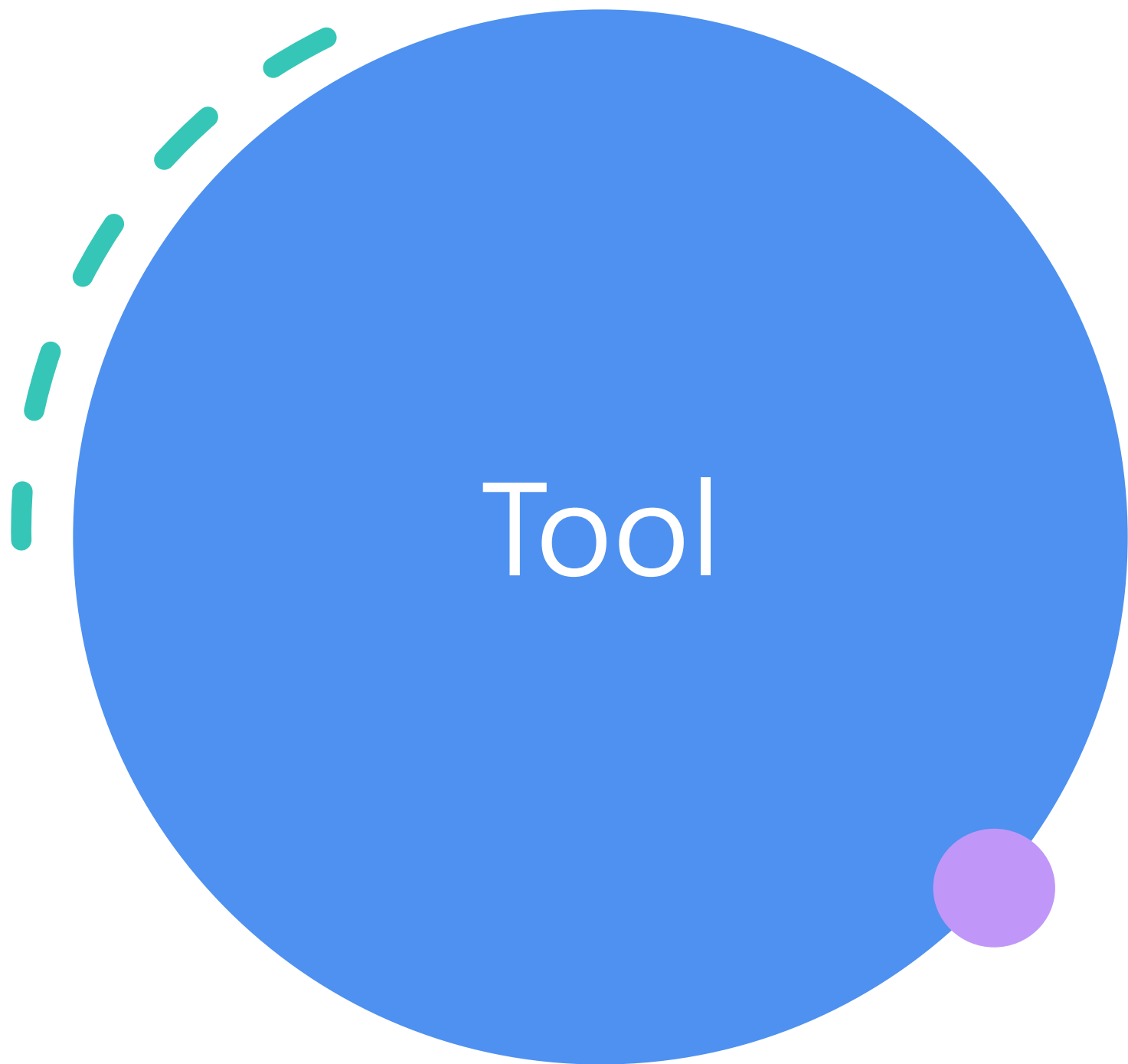
Professor : Chi-Yu Lin (林其禹)

Student : WEI YEN CHEN



# Outline

1. Tool
2. Program Design
3. Results and Discussion



# Tool



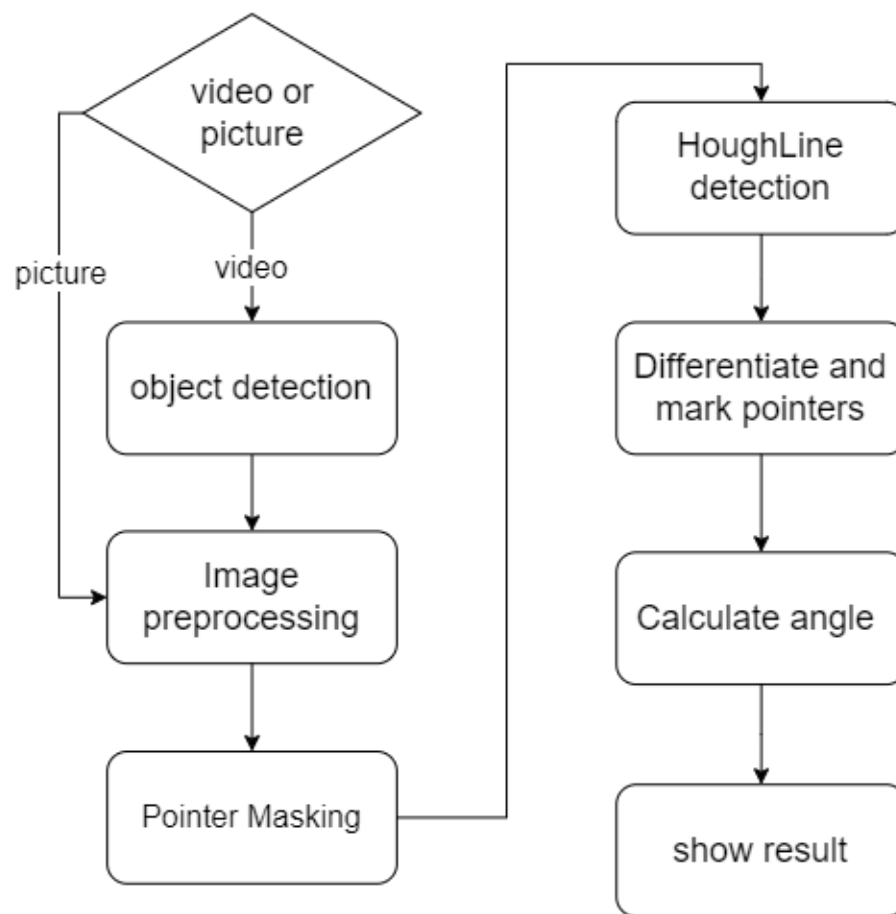
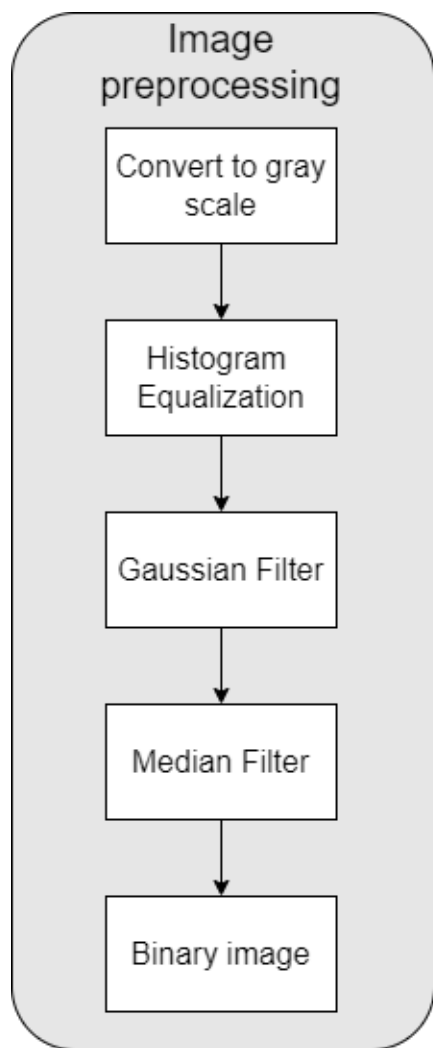
scikit-image  
image processing in python





# Program Design

# Workflow



# Steps

- Step1 Load picture or video
- Step2 Object detection (for video)
- Step3 Image preprocessing
- Step4 Pointer masking
- Step5 Hough Line detection
- Step6 Differentiate and mask pointers
- Step7 Calculate angle
- Step8 Show result

# Step1 Load picture or video

```
6 #Step1·影像讀取
7 mode=input("video or picture?")
8 if mode=="picture":
9     ...img=cv2.imread("./data/1.jpg",-1)
10    ...threshod=40
11    ...minLineLength=10
12    ...MaxLineGap=25
13 elif mode=="video":
14    ...cap=cv2.VideoCapture(0)
15    ...#cap=cv2.VideoCapture('./data/v1.mp4').....
16    ...cap.set(3,1280).....#width
17    ...cap.set(4,720).....#high
18    ...cap.set(10,70)
19    #*****model setup*****
20    ...thres=0.45# Threshold to detect object
21    ...classNames=[]
22    ...classFile="coco.names"
23    ...with open(classFile,'rt') as f:
24    ...    ...classNames=f.read().rstrip('\n').split('\n')
25
26    ...configPath='ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt'
27    ...weightsPath='frozen_inference_graph.pb'
28
29    ...net=cv2.dnn_DetectionModel(weightsPath,configPath)
30    ...net.setInputSize(320,320)
31    ...net.setInputScale(1.0/127.5)
32    ...net.setInputMean((127.5,127.5,127.5))
33    ...net.setInputSwapRB(True)
34    #*****
35    ...threshod=40
36    ...minLineLength=10
37    ...MaxLineGap=20
38    ...if not cap.isOpened():
39    ...    ...print("Cannot open camera")
40    ...    ...exit()
```

Img:





# Step2 Object detection

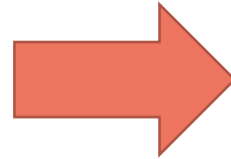
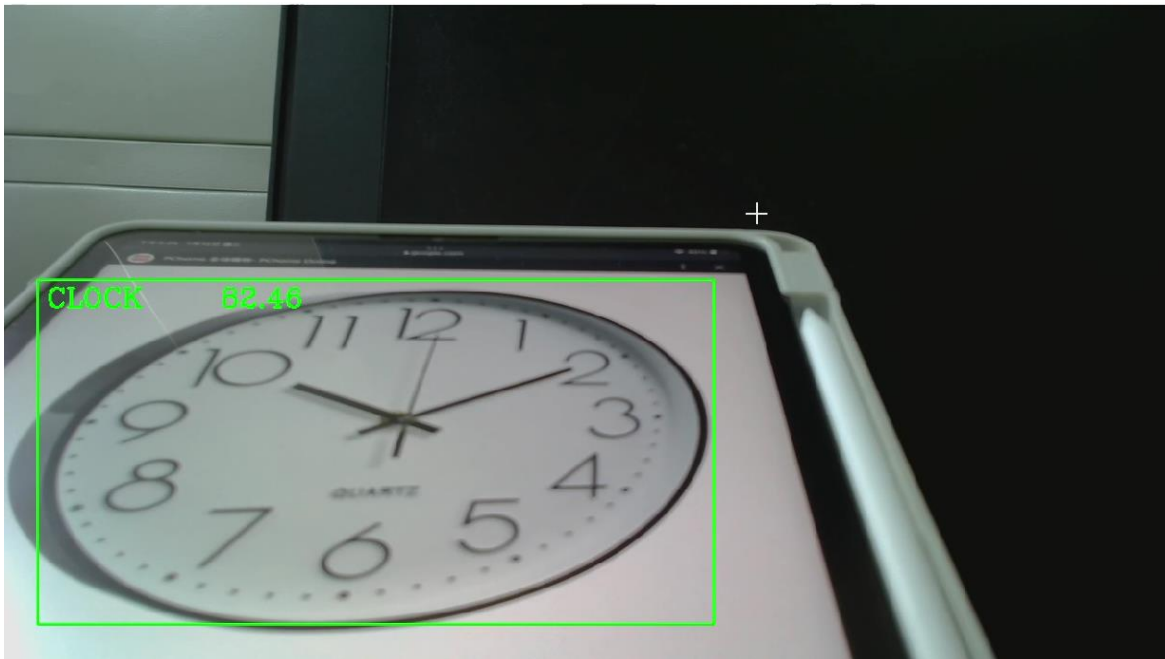
```
189 .....#物件偵測
190 .....classIds, confs, bbox = net.detect(img, confThreshold=thres)
191 .....if len(classIds) != 0:
192 .....    for classId, confidence, box in zip(classIds.flatten(), confs.flatten(), bbox):
193 .....        if classId == 85:
194 .....            cv2.rectangle(img, box, color=(0, 255, 0), thickness=2)
195 .....            cv2.putText(img, classNames[classId-1].upper(), (box[0]+10, box[1]+30),
196 .....                        cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 0), 2)
197 .....            cv2.putText(img, str(round(confidence*100, 2)), (box[0]+200, box[1]+30),
198 .....                        cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255, 0), 2)
199 .....            cv2.imshow("detection", img)
200 .....            if classId == 85:
201 .....                img0 = img0[box[1]:box[1]+box[3], box[0]:box[0]+box[2]]
202 .....                img1 = cv2.resize(img0, (600, 600))
203 .....            elif classId != 85:
204 .....                img1 = cv2.resize(img, (600, 600))
205 .....            elif mode == "picture":
206 .....                img1 = cv2.resize(img, (600, 600))
207 .....            img2 = img1.copy()
208 .....            img3 = img1.copy()
209 .....            img4 = img1.copy()
```

Select the ROI

Network :  
mobilenet\_v3

```
19 *****model setup*****
20 ....thres = 0.45 # Threshold to detect object
21 ....classNames = []
22 ....classFile = "coco.names"
23 ....with open(classFile, 'rt') as f:
24 ....    ....classNames = f.read().rstrip('\n').split('\n')
25
26 ....configPath = 'ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt'
27 ....weightsPath = 'frozen_inference_graph.pb'
28
29 ....net = cv2.dnn_DetectionModel(weightsPath, configPath)
30 ....net.setInputSize(320, 320)
31 ....net.setInputScale(1.0/127.5)
32 ....net.setInputMean((127.5, 127.5, 127.5))
33 ....net.setInputSwapRB(True)
34 *****
```

# Result



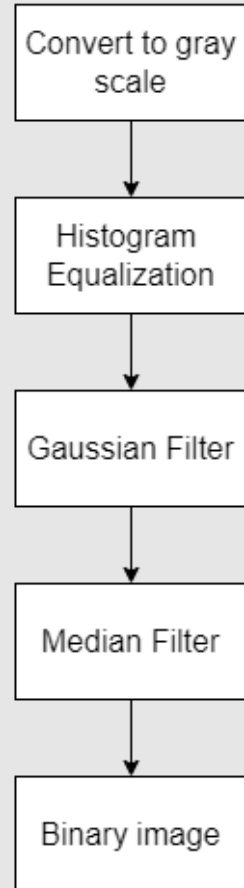
ROI



# Step3 Image preprocessing

```
42 def preprocessing(img):
43     ...#灰階
44     ...gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
45
46     ...#質方圖等化
47     ...clahe=cv2.createCLAHE(clipLimit=2)
48     ...clahe_img=clahe.apply(gray)
49
50     ...#高斯濾波 & 中值濾波器 (如果是網路照片就不使用)
51     ...if mode=="video":
52     ...    ...gaussian_img=cv2.GaussianBlur(clahe_img,(5,5),5)
53     ...    ...median_img=cv2.medianBlur(gaussian_img,5)
54     ...elif mode=="picture":
55     ...    ...gaussian_img=cv2.GaussianBlur(clahe_img,(5,5),0)
56     ...    ...median_img=gaussian_img
57
58     ...#二進值
59     ...ret,bin_img=cv2.threshold(median_img,90,255,cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
60
61     ...return bin_img
```

## Image preprocessing



# Result



Original image



Gray scale +  
Histogram Equalization

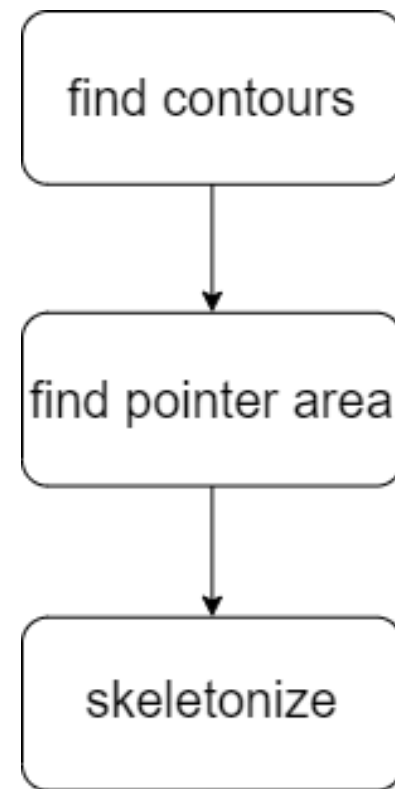


Gaussian Filter + Median  
Filter + convert to Binary  
image

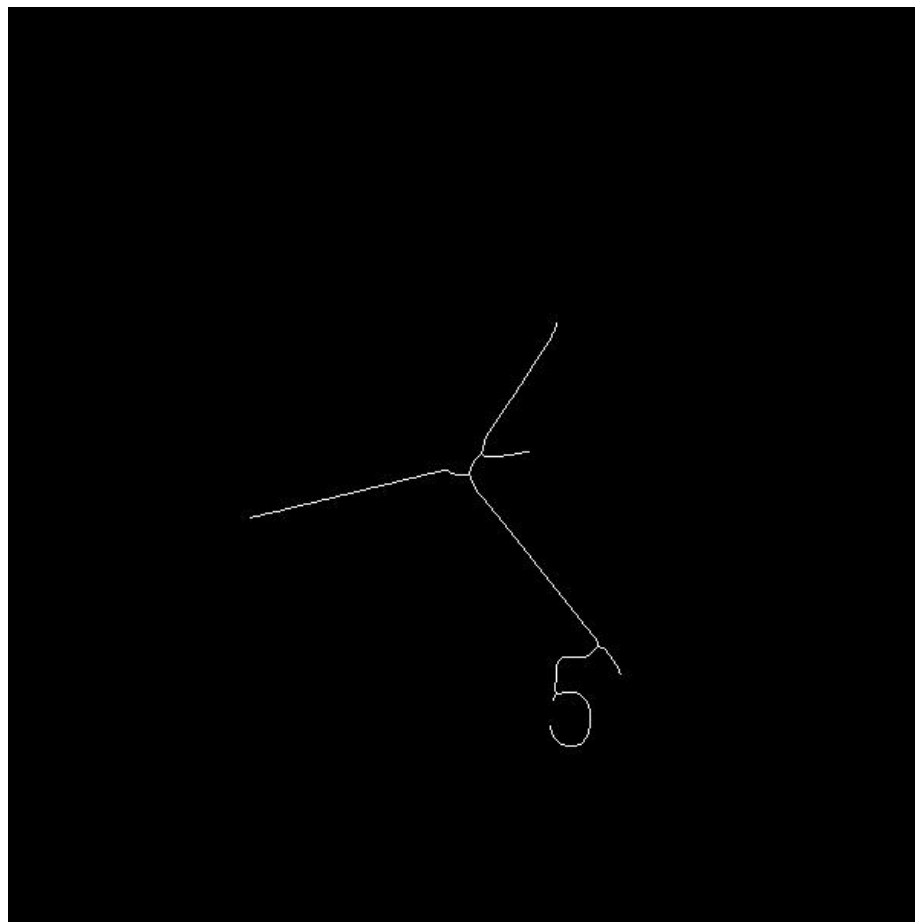
# Step4 Pointer masking

```
63 def get_contours(img):
64     ...#尋找輪廓
65     ...contours, hierarchy = cv2.findContours(bin_img, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
66
67     ...#計算面積，界在自訂的值內(2000<area<7000)才畫出輪廓
68     ...mask_img = np.zeros_like(bin_img)
69     ...for i, j in enumerate(contours):
70         ...area = int(cv2.contourArea(contours[i]))
71         ...#print("area=", area)
72         ...if area > 1500 and area < 7000:
73             ...cv2.drawContours(mask_img, contours, i, (255, 255, 255), -1)
74     ...return mask_img
```

```
212 #Step3 遮罩，提取指針
213 ...mask_img = get_contours(bin_img)
214 ...#骨架提取
215 ...mask_img[mask_img == 255] = 1
216 ...arms_thin = skeletonize(mask_img)
217 ...arms_thin = (255 * arms_thin).clip(0, 255).astype(np.uint8)
```



# Result



# Step5 Hough Line detection

```
76 def Houghline_detection():
77     ... linesP = cv2.HoughLinesP(arms_thin, 1, np.pi / 180, threshod, None, minLineLength, MaxLineGap)
78     ... print(linesP)
79     ... lines_length = []
80     ... p1 = []
81     ... p2 = []
82     ... if linesP is not None:
83         ... for i in range(0, len(linesP)):
84             ... l = linesP[i][0]
85             ... cv2.line(img3, (l[0], l[1]), (l[2], l[3]), (0, 255, 255), 2, cv2.LINE_AA)
86             ... p1.append(l[0])
87             ... p1.append(l[1])
88             ... p2.append(l[2])
89             ... p2.append(l[3])
90             ... length = int(math.sqrt((l[0]-l[2])**2+(l[1]-l[3])**2))
91             ... #length = int(math.dist(p1,p2))
92             ... lines_length.append(length)
93     ... return lines_length, linesP
```

# Result



Detected line

```
[[[158 333 325 292]]  
[[298 305 396 427]]  
[[295 305 344 232]]]
```



# Step6 Differentiate and mask pointers

```
95 def classify_pointer(lines_length, linesP):
96     ... #比較直線長度
97     ... t = lines_length
98     ... max_number = []
99     ... max_index = []
100     ... if linesP is not None:
101         ... for _ in range(3):
102             ... number = max(t)
103             ... index = t.index(number)
104             ... t[index] = 0
105             ... max_number.append(number)
106             ... max_index.append(index)
107         ... print(max_number)
108         ... print(max_index)
109
110     ... #劃出分針·時針
111     ... minute = []
112     ... hour = []
113     ... second = []
114     ... for i in range(4):
115         ... second.append(linesP[max_index[0]][0][i])
116         ... minute.append(linesP[max_index[1]][0][i])
117         ... hour.append(linesP[max_index[2]][0][i])
118
119     ... cv2.line(img4, (second[0], second[1]), (second[2], second[3]), (0, 255, 0), 3, cv2.LINE_AA)
120     ... cv2.line(img4, (minute[0], minute[1]), (minute[2], minute[3]), (0, 255, 255), 3, cv2.LINE_AA)
121     ... cv2.line(img4, (hour[0], hour[1]), (hour[2], hour[3]), (0, 0, 255), 3, cv2.LINE_AA)
122     ... print("second(G):", second)
123     ... print("minute(Y):", minute)
124     ... print("hour(R):", hour)
125     ... return second, minute, hour
126     ... else:
127         ... return 0, 0, 0
```

The longest : second head  
The second: minute head  
The third: hour head

# Result

```
[[[158 333 325 292]] 171  
 [[298 305 396 427]] 156  
 [[295 305 344 232]]] 87
```



```
[171, 156, 87]   Length  
[0, 1, 2]        Index of line  
second(G): [158, 333, 325, 292]  
minute(Y): [298, 305, 396, 427]  
hour(R): [295, 305, 344, 232]
```

# Step7 Calculate angle

```
129 def claculate_time(a,mode):
130     ...#print(a)
131     ...a[0] += a[0]-300
132     ...a[1] += -a[1]+300
133     ...a[2] += a[2]-300
134     ...a[3] += -a[3]+300
135     ...#print("校正:",a)
136     ...delta_x = a[2]-a[0]
137     ...delta_y = a[3]-a[1]
138     ...print("delta_x=",delta_x)
139     ...print("delta_y=",delta_y)
140     ...#判斷象限
141     ...if delta_y/delta_x > 0 and a[0] < -45:
142     ...     ...pi=math.pi
143     ...elif delta_y/delta_x < 0 and a[0] < -50 and a[2] < 35:
144     ...     ...pi=math.pi
145     ...else:
146     ...     ...pi=0
147     ...print("pi=",pi)
148     ...#計算角度
149     ...angle = math.degrees(math.atan2(delta_y,delta_x) + pi)
150     ...if angle < 0:
151     ...     ...angle=angle+360
152     ...print("arctan=",angle)
153     ...#轉換角度座標
154     ...angle_CCW90 = angle - 90
155     ...if angle_CCW90 < 0:
156     ...     ...angle_CCW90 = angle_CCW90 + 360
157     ...angle_clock = -angle_CCW90 + 360
158     ...#如果最後角度>360，扣一圈(360)
159     ...if angle_clock > 360:
160     ...     ...angle_clock=angle_clock-360
161     ...print("angle_clock=",angle_clock)
```

```
163     ...#判斷3,6,9,0
164     ...if angle_clock % 90 == 0:
165     ...     ...if delta_y > 0:
166     ...         ...angle_clock = 0
167     ...     ...elif delta_y < 0:
168     ...         ...angle_clock = 180
169     ...     ...elif delta_x < 0:
170     ...         ...angle_clock = 90
171     ...     ...elif delta_x > 0:
172     ...         ...angle_clock = 270
173     ...#計算時間
174     ...if mode == 1:
175     ...     ...time = math.floor(angle_clock / 6)
176     ...elif mode == 0:
177     ...     ...time = math.floor(angle_clock / 30)
178     ...     ...print("time=",time)
179     ...return time
```

# Step8 Show result

```
231 #Step7 顯示結果
232 ....cv2.imshow("bin_img",bin_img)
233 ....cv2.imshow('arms_thin', arms_thin)
234 ....cv2.imshow("Hough Line DetectionP", img3)
235 ....cv2.putText(img4, "time:%d:%d:%d"%(time_hour,time_minute,time_second),\
236 .....(50,50), cv2.FONT_HERSHEY_SIMPLEX,1,(180,0,0),1,cv2.LINE_AA)
237 ....cv2.imshow("report", img4)
```

time:1:23:42





# Results and Discussion

# Success case

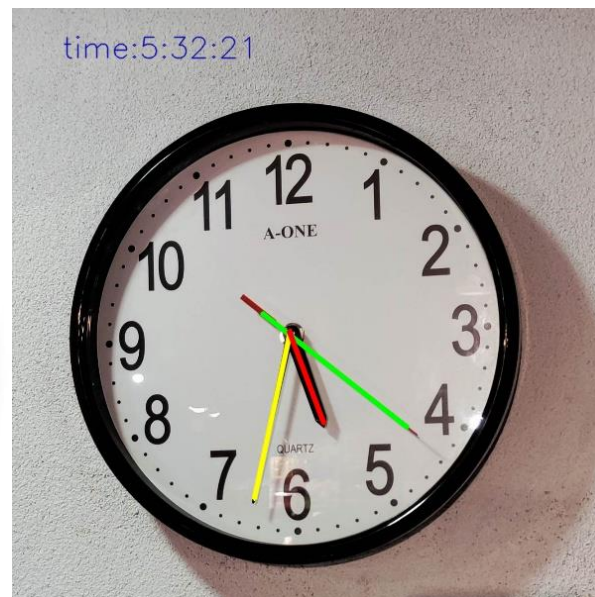
time:1:52:33



time:1:33:48



time:5:32:21

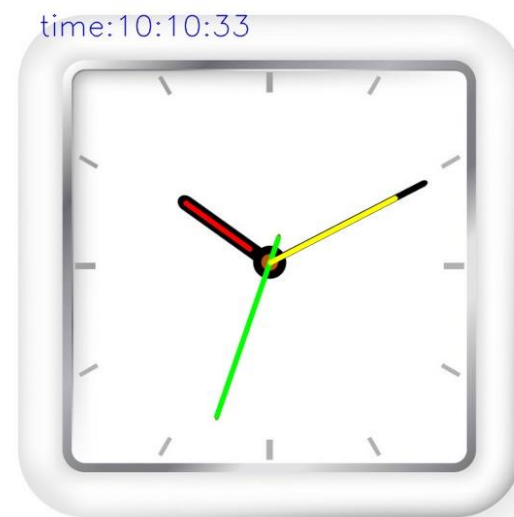


time:1:57:37

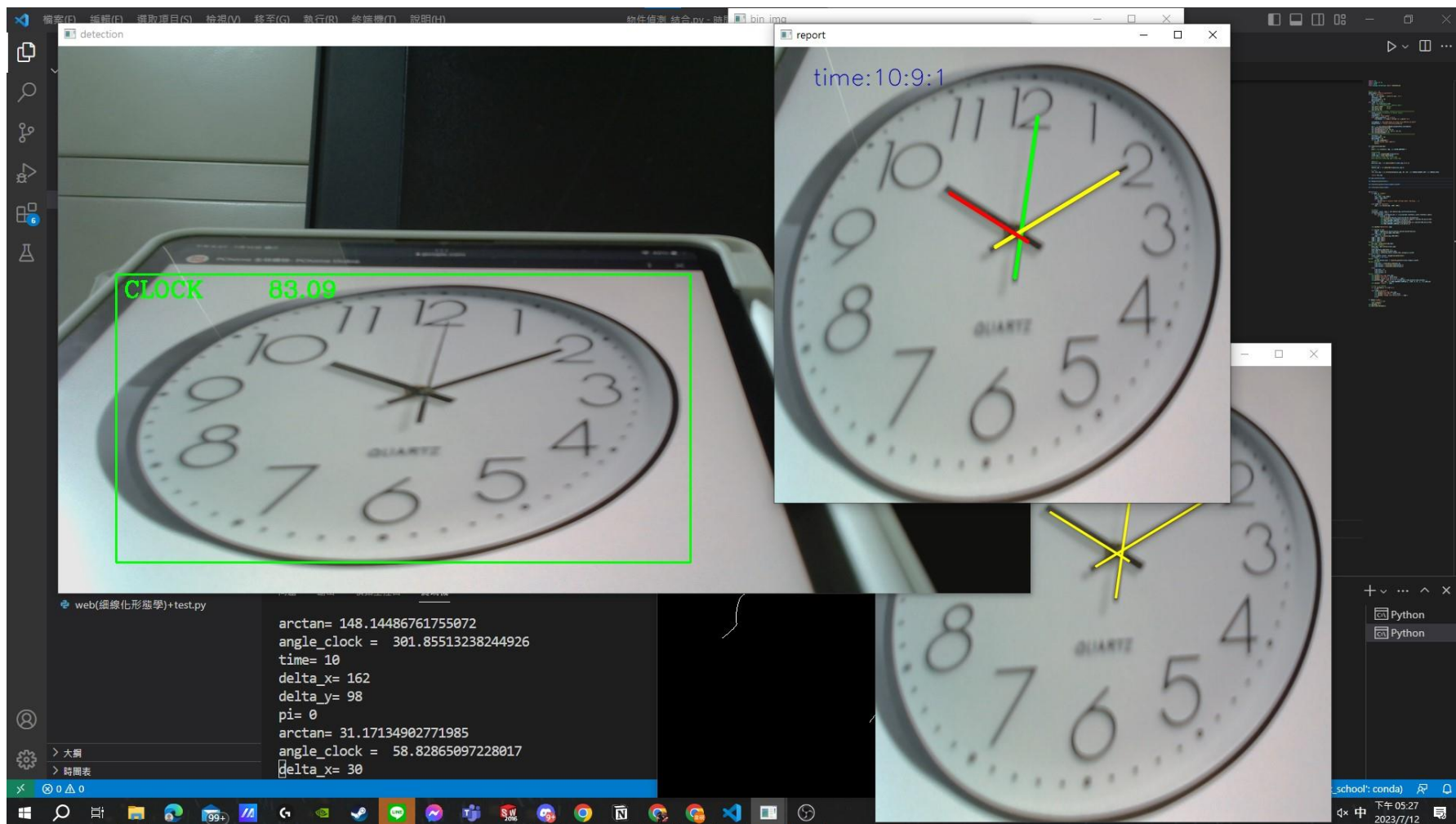




# Success case

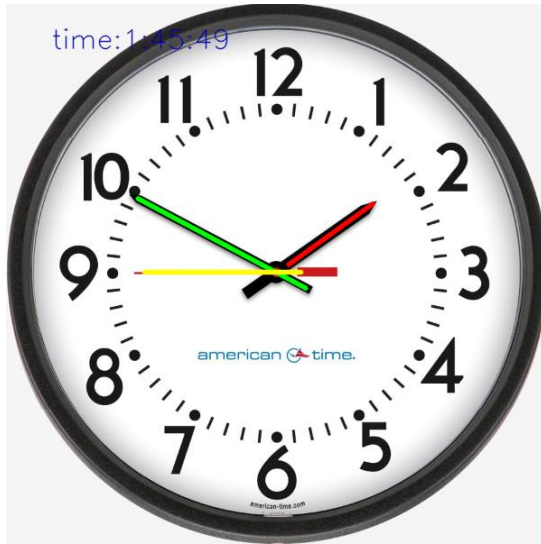


# Success case of video





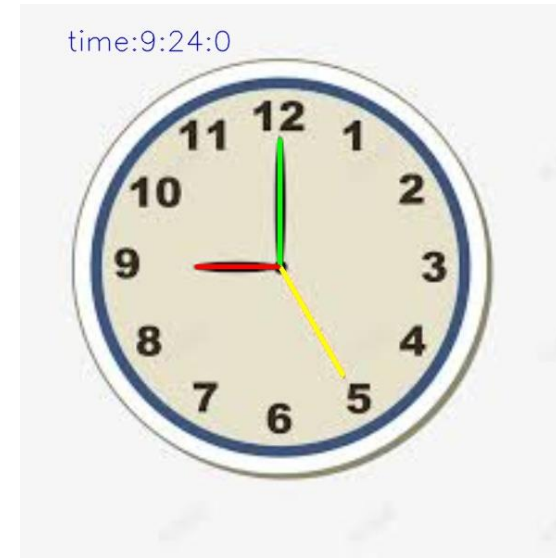
# Failure case



The hour hand and minute hand are reversed



Square clocks have different angles



The hour hand and minute hand have similar lengths.



The second hand has been eliminated due to preprocessing



# Future work

- Improve the preprocessing
- Improve the elimination of the second hand in some images
- Solve image rotation



Thanks for  
listening

# Code

```
1 import cv2
2 import numpy as np
3 import math
4 from skimage.morphology import skeletonize
5
6 #Step1 影像讀取
7 mode=input("video or picture?")
8 if mode == "picture":
9     img = cv2.imread( "./data/14.jpg", -1 )
10    threshod = 40
11    minLineLength = 10
12    MaxLineGap = 25
13
14 elif mode == "video":
15     cap = cv2.VideoCapture(0)
16     #cap = cv2.VideoCapture('./data/v1.mp4')
17     cap.set(3,1280)      #width 3 = cv2.CAP_PROP_FRAME_WIDTH
18     cap.set(4,720)      #high 4 = cv2.CAP_PROP_FRAME_HEIGHT
19     cap.set(10,70)
20     #*****model setup*****
21     thres = 0.45 # Threshold to detect object
22     classNames= []
23     classFile = "coco.names"
24     with open(classFile,'rt') as f:
25         classNames = f.read().rstrip('\n').split('\n')
26
27     configPath = 'ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt'
28     weightsPath = 'frozen_inference_graph.pb'
29
30     net = cv2.dnn_DetectionModel(weightsPath,configPath)
31     net.setInputSize(320,320)
32     net.setInputScale(1.0/ 127.5)
33     net.setInputMean((127.5, 127.5, 127.5))
34     net.setInputSwapRB(True)
35     #*****
36     threshod = 40      #40
37     minLineLength = 40 #10
38     MaxLineGap = 100   #20
39     if not cap.isOpened():
40         print("Cannot open camera")
41         exit()
```

# Code

```
43 def preprocessing(img):
44     #灰階
45     gray = cv2.cvtColor( img, cv2.COLOR_BGR2GRAY )
46
47     #質方圖等化
48     clahe = cv2.createCLAHE(clipLimit=2)
49     clahe_img = clahe.apply(gray)
50
51     #高斯濾波 & 中值濾波器 (如果是網路照片就不使用)
52     if mode == "video":
53         sigma = 5
54         kernel = 5
55     elif mode == "picture":
56         sigma = 0
57         kernel = 3
58     gaussian_img = cv2.GaussianBlur(clahe_img,(5,5),sigma)
59     median_img = cv2.medianBlur(gaussian_img,kernel)
60
61     #二進值
62     ret, bin_img = cv2.threshold(median_img, 90, 255, cv2.THRESH_BINARY_INV+ cv2.THRESH_OTSU)
63
64     return bin_img
65
66 def get_contours(img):
67     #尋找輪廓
68     contours, hierarchy = cv2.findContours(bin_img, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
69
70     #計算面積 · 界在自訂的值內(2000<area<7000)才畫出輪廓
71     mask_img = np.zeros_like(bin_img)
72     for i, j in enumerate(contours):
73         area = int(cv2.contourArea(contours[i]))
74         #print("area=",area)
75         if area>2200 and area<7000: #*****7000
76             cv2.drawContours(mask_img, contours, i, (255,255,255), -1)
77     return mask_img
```

# Code

```
66 def get_contours(img):
67     #尋找輪廓
68     contours, hierarchy = cv2.findContours(bin_img, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
69
70     #計算面積 · 界在自訂的值內(2000<area<7000)才畫出輪廓
71     mask_img = np.zeros_like(bin_img)
72     for i, j in enumerate(contours):
73         area = int(cv2.contourArea(contours[i]))
74         #print("area=",area)
75         if area>2200 and area<7000: *****7000
76             cv2.drawContours(mask_img, contours, i, (255,255,255), -1)
77     return mask_img
78
79 def Houghline_detection(img):
80     linesP = cv2.HoughLinesP(img, 1, np.pi / 180, threshod, None, minLineLength, MaxLineGap) #40 10 20' 40 10 25
81     print("minLineLength=",minLineLength)
82     #print(linesP)
83     lines_length=[]
84     p1 = []
85     p2 = []
86     if linesP is not None:
87         for i in range(0, len(linesP)):
88             l = linesP[i][0]
89             cv2.line(img3, (l[0], l[1]), (l[2], l[3]), (0,255,255), 2, cv2.LINE_AA)
90             p1.append(l[0])
91             p1.append(l[1])
92             p2.append(l[2])
93             p2.append(l[3])
94             length = int(math.sqrt((l[0]-l[2])**2+(l[1]-l[3])**2))
95             #length = int(math.dist(p1,p2))
96             lines_length.append(length)
97     return lines_length,linesP
```

# Code

```
99 def classify_pointer(lines_length, linesP):
100     #比較直線長度
101     t = lines_length
102     max_number = []
103     max_index = []
104     if linesP is not None:
105         for _ in range(3):
106             number = max(t)
107             index = t.index(number)
108             t[index] = 0
109             max_number.append(number)
110             max_index.append(index)
111         print(max_number)
112         print(max_index)
113
114     #劃出分針 時針
115     minute = []
116     hour = []
117     second = []
118     for i in range(4):
119         second.append(linesP[max_index[0]][0][i])
120         minute.append(linesP[max_index[1]][0][i])
121         hour.append(linesP[max_index[2]][0][i])
122
123     cv2.line(img4, (second[0], second[1]), (second[2], second[3]), (0, 255, 0), 3, cv2.LINE_AA)
124     cv2.line(img4, (minute[0], minute[1]), (minute[2], minute[3]), (0, 255, 255), 3, cv2.LINE_AA)
125     cv2.line(img4, (hour[0], hour[1]), (hour[2], hour[3]), (0, 0, 255), 3, cv2.LINE_AA)
126     print("second(G):", second)
127     print("minute(Y):", minute)
128     print("hour(R):", hour)
129     return second, minute, hour
130 else:
131     return 0, 0, 0
```

# Code

```
133 def claculate_time(a,mode):
134     #print(a)
135     a[0] = a[0]-300
136     a[1] = -a[1]+300
137     a[2] = a[2]-300
138     a[3] = -a[3]+300
139     #print("校正:",a)
140     delta_x = a[2]-a[0]
141     delta_y = a[3]-a[1]
142     print("delta_x=",delta_x)
143     print("delta_y=",delta_y)
144     #判斷象限
145     if delta_y/delta_x > 0 and a[0]<-45:
146         pi=math.pi
147     elif delta_y/delta_x < 0 and a[0]<-50 and a[2]<35:
148         pi=math.pi
149     else:
150         pi=0
151     print("pi=",pi)
152     #計算角度
153     angle = math.degrees(math.atan2(delta_y,delta_x) + pi)
154     if angle < 0 :
155         angle=angle+360
156     print("arctan=",angle)
157     #轉換角度座標
158     angle_CCW90 = angle - 90
159     if angle_CCW90 < 0:
160         angle_CCW90 = angle_CCW90 + 360
161     angle_clock = -angle_CCW90 + 360
162     #如果最後角度>360 · 扣一圈(360)
163     if angle_clock>360:
164         angle_clock=angle_clock-360
165     print("angle_clock = ",angle_clock)
167     #判斷3,6,9,0
168     if angle_clock % 90 == 0:
169         if delta_y > 0:
170             angle_clock = 0
171         elif delta_y < 0:
172             angle_clock = 180
173         elif delta_x < 0:
174             angle_clock = 90
175         elif delta_x > 0:
176             angle_clock =270
177     #計算時間
178     if mode==1:
179         time = math.floor(angle_clock / 6)
180     elif mode ==0:
181         time = math.floor(angle_clock / 30)
182         print("time=",time)
183     return time
```



# Code

```
185 while(True):
186     if mode == "video":
187         # 擷取影像
188         ret, img = cap.read()
189         img0 = img.copy()
190         if not ret:
191             print("Can't receive frame (stream end?). Exiting ...")
192             break
193         #物件偵測
194         classIds, confs, bbox = net.detect(img,confThreshold=thres)
195         if len(classIds) != 0 :
196             for classId, confidence,box in zip(classIds.flatten(),confs.flatten(),bbox):
197                 if classId == 85:
198                     cv2.rectangle(img,box,color=(0,255,0),thickness=2)
199                     cv2.putText(img,classNames[classId-1].upper(),(box[0]+10,box[1]+30),
200                                 cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)
201                     cv2.putText(img,str(round(confidence*100,2)),(box[0]+200,box[1]+30),
202                                 cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)
203                 cv2.imshow("detection",img)
204                 if classId == 85:
205                     img0 = img0[box[1]:box[1]+box[3],box[0]:box[0]+box[2]]
206                     img1 = cv2.resize(img0,(600,600))
207                 elif classId != 85:
208                     img1 = cv2.resize(img,(600,600))
209             elif mode == "picture":
210                 img1 = cv2.resize(img, (600, 600))
211             img2 = img1.copy()
212             img3 = img1.copy()
213             img4 = img1.copy()
214             #Step2 圖像預處理
215             bin_img = preprocessing(img1)
216             #Step3 遮罩・提取指針
217             mask_img = get_contours(bin_img)
218             #骨架提取
219             mask_img[mask_img==255] = 1
220             arms_thin = skeletonize(mask_img)
221             arms_thin = (255*arms_thin).clip(0,255).astype(np.uint8)
222             #Step4 直線偵測
223             lines_length,linesP = Houghline_detection(arms_thin)
224             if linesP is not None:
225                 #Step5 區分指針
226                 second,minute,hour = classify_pointer(lines_length,linesP)
```

# Code

```
227 #Step6 判斷時間
228     time_hour = claculate_time(hour,0)
229     time_minute = claculate_time(minute,1)
230     time_second = claculate_time(second,1)
231     else:
232         time_hour = 0
233         time_minute = 0
234         time_second = 0
235 #Step7 顯示結果
236     cv2.imshow("bin_img",bin_img)
237     cv2.imshow('arms_thin', arms_thin)
238     cv2.imshow( "Hough Line DetectionP ", img3 )
239     cv2.putText(img4, "time:%d:%d:%d" %(time_hour,time_minute,time_second), \
240                 (50, 50), cv2.FONT_HERSHEY_SIMPLEX,1, (180, 0, 0), 1, cv2.LINE_AA)
241     cv2.imshow( "report ", img4 )
242
243     # 按下 q 鍵離開迴圈
244     if cv2.waitKey(1) == ord('q'):
245         break
246     elif mode=="picture":
247         cv2.imshow("bin_img",bin_img)
248         cv2.imshow('arms_thin', arms_thin)
249         cv2.imshow( "Hough Line DetectionP ", img3 )
250         break
251
252 if mode=="video":
253     # 釋放該攝影機裝置
254     cap.release()
255 cv2.waitKey( 0 )
256 cv2.destroyAllWindows()
```