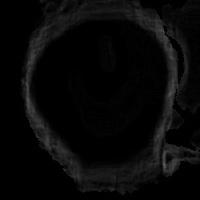
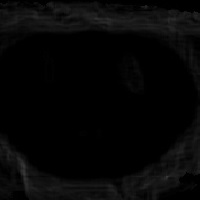


1. 影像灰階處理:
   1. 讀取影像圖片 cvCapture
   2. 再利用函式把圖片轉成灰階
2. 取邊界
   1. 高斯模糊 GaussianBlur
   2. 再利用函式庫裡的Sobel分別對x、y取邊界
3. 背景處理
   1. 利用第一張灰階圖片當背景，之後每十張比較一次，每一點都有一個最大值跟最小值，只要灰階圖片該點像素大於十張圖片中該點最大值或小於十張圖片中該點最小值時更新最小值跟最大值，最後算出最大值跟最小值的差
   2. 如果小於更新門檻值(10)，背景就會更新
   3. 變數說明:
      1. 該十張圖片像素值最大的點
      2. 該十張圖片像素值最小的點
      3. 十張圖片該點平均值
      4. 背景更新後該點的像素值
      5. 更新背景時，背景所佔的比例(0.3)
      6. 原本背景該點的像素值
4. 量化差異
   1. 利用公式把差異量化
      1. 代表量化後的像素
      2. 代表該點座標
      3. 代表灰階圖片跟背景圖片該點像素的差異
      4. 代表灰階圖片跟背景圖片像素差異的最大值
      5. 代表灰階圖片跟背景圖片像素差異的最小值
5. 膨脹侵蝕
   1. 先用opencv裡的erode侵蝕過濾掉雜訊
   2. 再用dilate膨脹把輪廓再接起來
6. 二值化
   1. 我們設定一個門檻值(20)，灰階圖片該點的像素，如果大於門檻值，像素變為255，反之，則變成0。
7. 取得前景邊界圖
   1. 利用opencv函式庫中的把二值化的結果跟邊界圖片結合即成為前景邊界圖。
8. 頭部樣板
   1. 樣板的大小限定在200\*200
      1. 有標準圓形
      2. 頭+肩膀
9. 頭部偵測
   1. 利用頭部樣板去偵測是否為人頭，
   2. 再用 opencv 裡的matchTemplate 比對，該圖上每一點的像素代表灰階圖片跟頭部樣本的比對符合值，越符合值就越高
   3. 利用 threshold 設門檻值把比對結果圖二值化
   4. 設定一個相似度門檻值，若相似度大於此門檻，才確認為人頭
10. 預測位置
    1. 以所偵測到的人頭座標，向右邊及下方各取5個點，再將這25個點的像素質存於陣列中，接著利用下方公式計算相似度來判別是否為同一點

r =

c(x, y) 目前所取的點像素

p(x, y) 前一張所取的點像素

當 r 越小時，則兩點相似度越高

1. 接著將偵測範圍設定以加快運算速度
   * 1. 左右設為樣板寬度的1/2
     2. 上下設為與樣板長度相同
     3. 虛線內為有效的偵測範圍
2. 計算人數
   1. 設變數 life 為偵測到的點的生命週期，每一張圖片增加 life 次數1次
   2. 設變數 detected\_life 為偵測到的點又被追蹤次數，每追蹤一次 detected\_life 次數加1次
   3. 降低誤判:
      1. 當 life-dectected\_life > 4 時，認定此點為誤判，清除此點資料
      2. 當座標過於接近時，以life大者為正確追蹤到的點，life小者於以清除
      3. 設 bool 變數 count 為判別是否重複偵測
   4. 當 count = false 且 點通過中線時，再以前後座標判別方向，以計算人數