Image Processing (using OpenCV-Python)

2016年**4**月**12**日 星期二

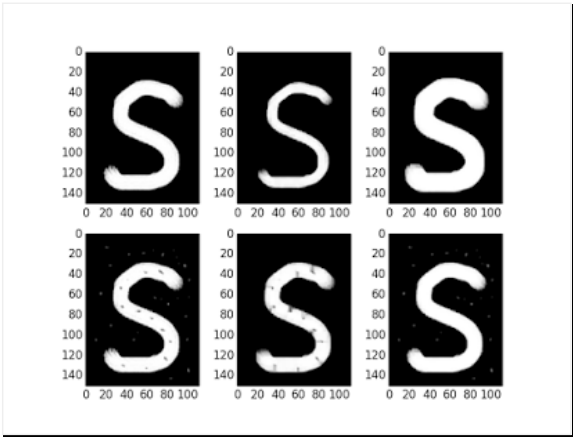
型態學轉換

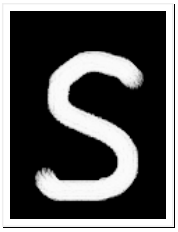
此範例說明如何使用OpenCV的型態學轉換

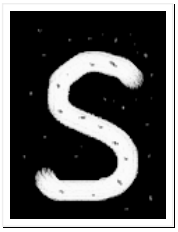
程式碼

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('d:\images\s.png',0)
kernel = np.ones((5,5),np.uint8)
erosion = cv2.erode(img,kernel,iterations = 1)  # 只執行一次
dilation = cv2.dilate(img,kernel,iterations = 1)
img2 = cv2.imread('d:\images\s1.png',0)
# opening 相當於先做erosion再做dilation，可移除雜訊
opening = cv2.morphologyEx(img2, cv2.MORPH_OPEN, kernel)
# closing 相當於先做dilation再做erosion，可將物件中的小洞補起來
closing = cv2.morphologyEx(img2, cv2.MORPH_CLOSE, kernel)
plt.subplot(2,3,1),plt.imshow(img,'gray')
plt.subplot(2,3,2),plt.imshow(erosion,'gray')
plt.subplot(2,3,3),plt.imshow(dilation,'gray')
plt.subplot(2,3,4),plt.imshow(img2,'gray')
plt.subplot(2,3,5),plt.imshow(opening,'gray')
plt.subplot(2,3,6),plt.imshow(closing,'gray')
plt.show()
```

執行結果







說明

1. s.png和s1.png兩張圖形顯示如上。
2. 可利用getStructuringElement產生所需要的Structuring Element,
例如下列程式產生相同的kernel

```
#kernel = np.ones((5,5),np.uint8)
kernel = cv2.getStructuringElement(cv2.MORPH_RECT,(5,5))
```

3. 下列程式可分別產生橢圓形與十字之Structuring Element

```
cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(5,5))
cv2.getStructuringElement(cv2.MORPH_CROSS,(5,5))
```

關於我自己

Michael Su

追蹤 0

檢視我的完整簡介

2016年**4**月**8**日 星期五

Thresholding

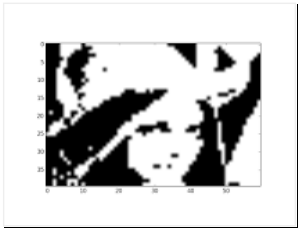
此範例說明如何使用簡單的thresholding和Otsu's thresholding

程式碼

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('d:\images\lena6040.bmp',0)

# BINARY  th=127, maxval=255,  des=maxval if src>th; otherwise, des = 0
ret,thresh1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)

# BINARY_INV  th=127, maxval=255,  des=0 if src>th; otherwise, des = maxval
ret,thresh2 = cv2.threshold(img,127,255,cv2.THRESH_BINARY_INV)

# TRUNC  th=127, maxval=255,  des=th if src>th; otherwise, des = src
ret,thresh3 = cv2.threshold(img,127,255,cv2.THRESH_TRUNC)

# TOZERO  th=127, maxval=255,  des=src if src>th; otherwise, des = 0
ret,thresh4 = cv2.threshold(img,127,255,cv2.THRESH_TOZERO)

# TOZERO_INV  th=127, maxval=255,  des=0 if src>th; otherwise, des = src
ret,thresh5 = cv2.threshold(img,127,255,cv2.THRESH_TOZERO_INV)

titles = ['Original Image','BINARY','BINARY_INV','TRUNC','TOZERO','TOZERO_INV']
images = [img, thresh1, thresh2, thresh3, thresh4, thresh5]

for i in xrange(6):
    plt.subplot(2,3,i+1),plt.imshow(images[i],'gray')
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])

plt.show()

# Otsu's thresholding
# 回傳值ret為最佳之門檻值
# 若以ret來當Canny edge detection的high threshold,
# 0.5*ret當low threshold, 通常可以得到很好之邊緣偵測結果
ret, th2 = cv2.threshold(img, 0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
print ret
plt.imshow(th2,'gray')
plt.show()
```

執行結果


```
>>>
118.0
>>>
```



張貼者： Michael Su 於 上午3:49 沒有留言:

2016年**4**月**6**日 星期三

Histograms - 1

此範例說明如何找到圖形的histogram, 並且畫出來。

程式碼

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Read an image
img = cv2.imread('d:\images\lena.bmp')
img_g = cv2.imread('d:\images\lena.bmp',0) # grayscale image

# 直接利用Matplotlib之pyplot的hist函式畫出histogram
plt.hist(img_g.ravel(),256,[0,256]); plt.show()

color = ('b', 'g', 'r')
for i,col in enumerate(color):     #i的值為0,1,2; col的值為'b', 'g', 'r'
    histr =cv2.calcHist([img],[i],None,[256],[0,256])
    plt.plot(histr,color=col)          #分別畫出b, g, r平面的histogram
    plt.xlim([0,256])                  #限制x軸的範圍為0~255

plt.show()
```

說明

cv2.calcHist(images, channels, mask, histSize, ranges[, hist[, accumulate]])

參數

images: 影像來源, uint8 or float32格式, 必須以中括號的格式給定"[img]"

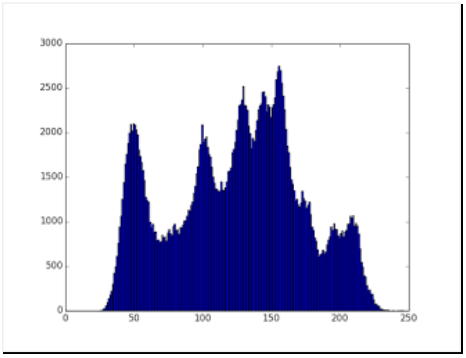
channels: 灰階影像為[0], 彩色影像為[0], [1], or [2]分別求b, g, r的histogram

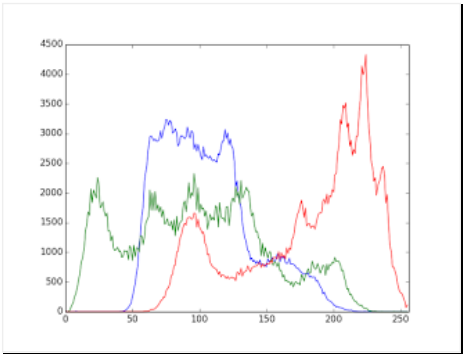
mask: 整張影像設定為None, 部分影像可設定區域

histSize: 即BIN值, 若是全部值則設定為"[256]", 若只要16個值(0~15, 16~31, 32~47, ...)則設定為"[16]"

ranges: intensity值的範圍, 若是所有的灰階值則設定為"[0,256]"

執行結果





張貼者： Michael Su 於 上午5:57 沒有留言:

2016年**4**月**2**日 星期六

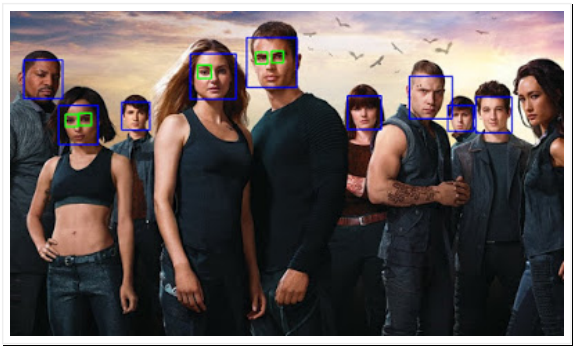
Face detection

此範例說明如何利用Haar cascade進行人臉偵測及眼睛偵測

程式碼

```
import numpy as np
import cv2

# 利用OpenCV已訓練好之人臉和眼睛之分類器
# 這兩個分類器存放在OpenCV3.0.0\sources\data\haarcascades
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')

# 讀取divergent電影之宣傳海報檔
img = cv2.imread('d:\images\divergent2.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
# 人臉偵測
faces = face_cascade.detectMultiScale(gray, 1.3, 5)
for (x,y,w,h) in faces:
    cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)  # 畫藍色矩形
    roi_gray = gray[y:y+h, x:x+w]                   # 設定人臉之ROI位置
    roi_color = img[y:y+h, x:x+w]
    eyes = eye_cascade.detectMultiScale(roi_gray) # 眼睛偵測
    for (ex,ey,ew,eh) in eyes:
        cv2.rectangle(roi_color,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)

cv2.imshow('img',img)
cv2.imwrite('d:\images\divergent_dt.jpg',img)       # 儲存影像
cv2.waitKey(0)
cv2.destroyAllWindows()
```

執行結果



由上圖可知, 最右邊的人臉並沒有偵測到
只有三個人的眼睛有被偵測到, 且女主角的眼睛只偵測到右眼
因此, 人的頭和眼睛面積太小, 可能會偵測不到

張貼者： Michael Su 於 上午4:54 沒有留言:

2016年**3**月**29**日 星期二

K-means

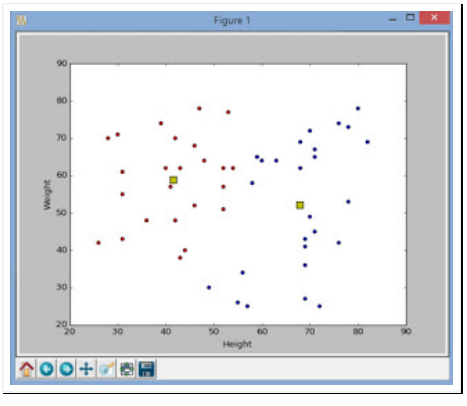
此範例說明如何利用K-means做分群

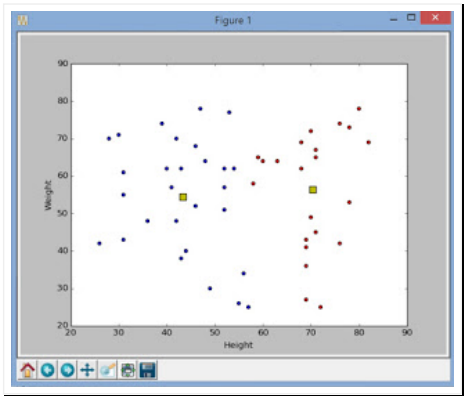
程式碼

```
import numpy as np
import cv2
from matplotlib import pyplot as plt

# 1D case
x = np.random.randint(25,100,25)
y = np.random.randint(175,255,25)   #產生25個隨機變數, 每個數的值介於(175,255)之間
z = np.hstack((x,y))
```

```
z = z.reshape((50,1))                    #合併成一個序列
z = np.float32(z)                        #轉成浮點數, 以便輸入至kmeans
plt.hist(z,256,[0,256]),plt.show()

# Define criteria = ( type, max_iter = 10 , epsilon = 1.0 )
# max_iter 或epsillon任一條件滿足即停止
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0)

# Set flags (Just to avoid line break in the code)
flags = cv2.KMEANS_RANDOM_CENTERS

# Apply KMeans (在此K=2, 分兩群)
compactness,labels,centers = cv2.kmeans(z,2,None,criteria,10,flags)

A = z[labels==0]
B = z[labels==1]

# Now plot 'A' in red, 'B' in blue, 'centers' in yellow
plt.hist(A,256,[0,256],color = 'r')
plt.hist(B,256,[0,256],color = 'b')
plt.hist(centers,32,[0,256],color = 'y')
plt.show()

# ----------------------------------------------------
# 2D case
# ----------------------------------------------------
X = np.random.randint(25,80,(25,2))  #產生25個2維的隨機變數, 每個數的值介於(25,80)之間
Y = np.random.randint(40,85,(25,2))
Z = np.vstack((X,Y))

# convert to np.float32
Z = np.float32(Z)

# apply kmeans()
# 執行5次, 以便觀察flags因設定成KMEANS_RANDOM_CENTERS的分類結果
for i in range(5):
    ret,label,center=cv2.kmeans(Z,2,None,criteria,1,flags)

    # Now separate the data, Note the flatten()
    A = Z[label.ravel()==0]
    B = Z[label.ravel()==1]

    # Plot the data
    plt.scatter(A[:,0],A[:,1])
    plt.scatter(B[:,0],B[:,1],c = 'r')
    plt.scatter(center[:,0],center[:,1],s = 80,c = 'y', marker = 's')
    plt.xlabel('Height'),plt.ylabel('Weight')
    plt.show()
```

執行結果 **(2D case)**





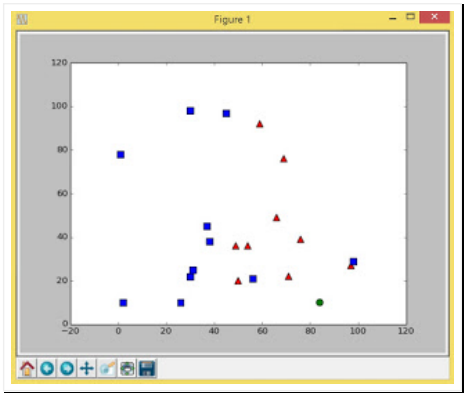
K-Nearest Neighbour

此範例說明如何利用kNN做分類

程式碼

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Feature set containing (x,y) values of 20 known/training data
# 每個random number的值介於(0,100), 20個2維的數值
trainData = np.random.randint(0,100,(20,2)).astype(np.float32)

# Labels each one either Red or Blue with numbers 0 and 1
# 任意指定20個值為0 或1
responses = np.random.randint(0,2,(20,1)).astype(np.float32)

# Take Red families and plot them; 0 表示紅色
red = trainData[responses.ravel()==0]
plt.scatter(red[:,0],red[:,1],80,'r','^')

# Take Blue families and plot them; 1 表示藍色
blue = trainData[responses.ravel()==1]
plt.scatter(blue[:,0],blue[:,1],80,'b','s')

# 產生一個值介於(0,100) 的2維數值
newcomer = np.random.randint(0,100,(1,2)).astype(np.float32)
plt.scatter(newcomer[:,0],newcomer[:,1],80,'g','o')

knn = cv2.KNearest()
knn.train(trainData,responses)              # k = 3, 找出距離newcomer最近的3個點
ret, results, neighbours ,dist = knn.find_nearest(newcomer, 3)

print "result: ", results,"\n"
print "neighbours: ", neighbours,"\n"
print "distance: ", dist

plt.show()
```

執行結果




```
result:  [[ 0.]]
```

neighbours: [[0. 0. 1.]]

distance: [[313. 458. 557.]]

由結果可知, newcomer(綠色點)被歸類為紅色(result: 0), 其中最近的k(k=3)個neighbors, 有2個紅色1個藍色。事實上, k可設成偶數, 例如k=4, 若有2個紅色2個藍色距離最近, 則會根據加總後的距離來判斷最後的結果, 最少者為其分類結果

2016年**3**月**28**日 星期一

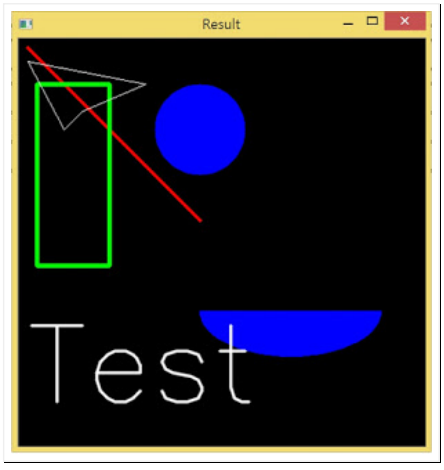
畫圖函式

此範例說明如何利用OpenCV的畫圖函式畫圖。

程式碼

```
import numpy as np
import cv2

# Create a black image
img = np.zeros((450,450,3), np.uint8)

# Draw a diagonal red line with thickness of 3 px
img = cv2.line(img,(10,10),(200,200),(0,0,255),3)  # (B, G, R)

# Drawing rectangle
img = cv2.rectangle(img,(20,50),(100,250),(0,255,0),3)

# Drawing circle
img = cv2.circle(img,(200,100), 50, (255,0,0), -1)

# Drawing ellipse
img = cv2.ellipse(img,(300,300),(100,50),0,0,180,255,-1)

# Drawing polygon
pts = np.array([[10,25],[140,50],[70,80],[50,100]], np.int32)
pts = pts.reshape((-1,1,2))     # ROWSx1x2    在此ROWS等於4, 有4個頂點, 格式必須是int32
img = cv2.polylines(img,[pts],True,(255,255,255))   # white

# Adding text
font = cv2.FONT_HERSHEY_SIMPLEX
cv2.putText(img,'Test',(10,400), font, 4,(255,255,255),2,cv2.LINE_AA)

cv2.imshow('Result',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

執行結果



存取Video

此範例說明如何利用OpenCV的模組, 存取Video。

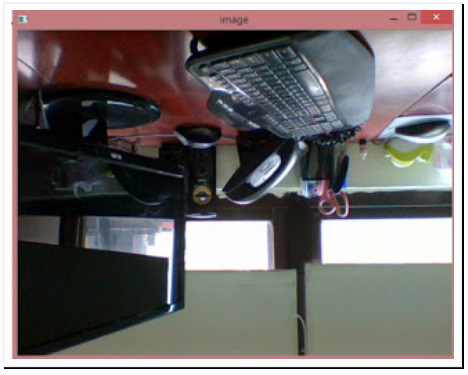
程式碼

```
import cv2
import numpy as np

# Read from camera
cap = cv2.VideoCapture(0)
#  Read from file
#  cap = cv2.VideoCapture('akiyo.avi')

# Define the codec and create VideoWrither object
fcc = cv2.VideoWriter_fourcc(*'MJPG')  # MJPG, XVID, X264, WMV1, WMV2
out = cv2.VideoWriter('test1.avi',fcc,20.0,(640,480))

while (cap.isOpened()):
    ret, frame = cap.read()
    if ret==True:
        frame = cv2.flip(frame,0)     # 畫面上下顛倒
        out.write(frame)                  # 將frame儲存至test1.avi
        cv2.imshow('image',frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
    else:
        break

#Release everything
cap.release()
out.release()
cv2.destroyAllWindows()
```

說明

利用FourCC指定編解碼的格式, 並利用VideoWriter存檔

執行結果



存取images

此範例說明如何利用OpenCV的模組, 存取影像。

程式碼

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Read an image
img = cv2.imread('d:\images\lena.bmp')
img_g = cv2.imread('d:\images\lena.bmp',0) # grayscale image

# Display an image
cv2.imshow('gray image',img_g)
cv2.waitKey(0)
```

```
cv2.destroyAllWindows()


# Use Matplotlib
## Color image loaded by OpenCV is in BGR mode
## But Matplotlib displays in RGB mode
b,g,r = cv2.split(img)
img2 = cv2.merge([r,g,b])
plt.subplot(121);plt.imshow(img) # expects distorted color
plt.subplot(122);plt.imshow(img2) # expect true color
plt.show()


# Write an image
cv2.imshow('bgr image',img) # expects true color
cv2.imshow('rgb image',img2) # expects distorted color
k = cv2.waitKey(0)
if k == 27:                 # wait for ESC key to exit
    cv2.destroyAllWindows()
elif k == ord('s'):       # wait for 's' key to save and exit
    cv2.imwrite('d:\images\lena_t.jpg',img)
    cv2.destroyAllWindows()
```

說明

1. 先利用import載入cv2, numpy(底下用np表示), 和matplotlib.pyplot(底下用plt表示)模組
2. 利用cv2.imread讀取彩色影像lena.bmp
3. 利用cv2.imread讀取lena.bmp的灰階值
4. 顯示img_g影像, 並且等待按鍵
5. 取出img之[b, g, r]成分, 並且組成[r, g, b]之img2影像以便利用Matplotlib顯示影像
6. 利用subplot分割畫面, 顯示結果如下
7. 利用cv2.imshow分別顯示img, 和img2
8. 等待按鍵, 並將值存在變數k
9. 當按鍵是Esc時, 則exit, 否則, 將img另存成新檔lena_t.jpg

執行結果