Assignment1 圖像旋轉

## 方法

### 旋轉整張圖像

方法:利用拉桿取得角度後，依角度旋轉。

#### 用拉桿取得角度

使用**cv::**createTrackbar(拉桿名稱, 放在哪個視窗, 值, 最大值, 回調函數, 傳給回調函數的值)

會呼叫回調函數rotateImage，可在函數裡進行旋轉圖片。把值與會用到的圖以全域變數宣告可以減少傳來傳去的麻煩。

#### 在回調函數中旋轉圖片

宣告一個mat型態的旋轉矩陣rotated。

先找到選轉中心(圖高、寬的一半)，與拉桿取得的值得到一個選轉矩陣。

使用cv::getRotationMatrix2D(選轉中心,角度,縮放)

再使用cv::warpAffine(輸入圖像,輸出圖像,旋轉矩陣,輸出圖像的尺寸,插值方法(默認為線性插值),像素外推法（默認為 BORDER\_CONSTANT）,border 值(默認為 0))

### 旋轉內切原圖像

方法:切下內切圓 > 使用拉桿選轉內切圓(與(a)旋轉圖片方法相同) > 把旋轉後的圓形貼回原圖片上

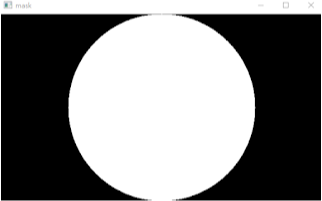
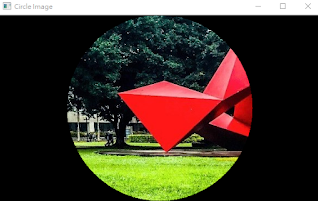
#### 裁切成圓形圖片

用圓形的mask遮罩留下中心內切圓區域:

先找出圓心(圖中心)，再找出半徑(圖2邊長較短邊)，使用

cv::circle(圖像,圓心坐標,圓形的半徑,線條的顏色,線條的粗細(正數>圓的線條的粗細程度，負數>填充),線條的類型,圓心坐標點和半徑值的小數點位數)

在大小跟原圖一樣的全黑圖上畫出白色圓形來做遮罩，再用img1.copyTo(img2,mask)製作剪下內切圓，img2得到的是img1被mask掩蓋後的圖，mask中黑色是被遮蓋的地方。

遮罩圓 內切圓圖

#### 使用拉桿選轉內切圓(與(a)旋轉圖片方法相同)

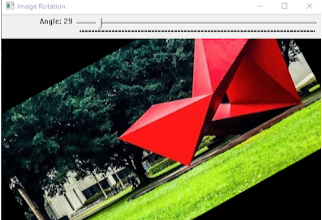
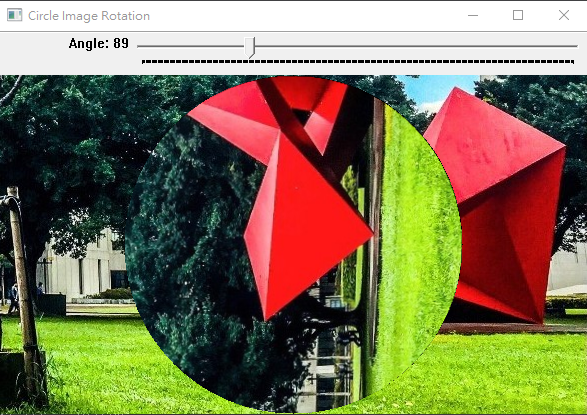
#### 把旋轉後的圓形貼回原圖片上

使用img1.copyTo(img2,mask)，img2是原圖的複製(img.clone())，img1是上一步驟得到的旋轉內切圓，mask是上上步驟使用到的圓形mask。

## 程式碼



## 成果

成果a 成果b

線條自畫像 (二值化與邊緣偵測 Image Thresholding and Edge Detection)

## 方法

### 利用 Sobel Operators 偵測並輸出邊緣成分圖

1. 用Sobel函數計算x方向和y方向的梯度值

2. 將圖像轉換為8位無符號整型圖像，否則無法顯示

3. 將兩個方向的結果進行加權合併

### 設計一個類似素描線條的自畫像圖案

1. 高斯模糊

2. 閾值化，去掉一些小線條細節

3. 黑白轉換

## 程式碼



## 成果

****

 離散傅立葉轉換 DFT 練習

## 方法

### a.圖像轉換至頻域

擴展圖像矩陣，為2，3，5的倍數時運算速度快。創建一個雙通道矩陣planes，用來儲存複數的實部與虛部。從多個單通道數組中創建一個多通道數組:transform\_image。函數Merge將幾個數組合併為一個多通道陣列，即輸出數組的每個元素將是輸入數組元素的級聯。在進行傅立葉變換。

### b.將頻譜大小與相位角度以灰階 256 色圖像方式呈現出

計算複數的幅值與相位。得到的頻譜圖與相位圖數級過大，不好顯示，因此轉換，剪切和重分佈幅度圖像限，使原點位於圖像中心

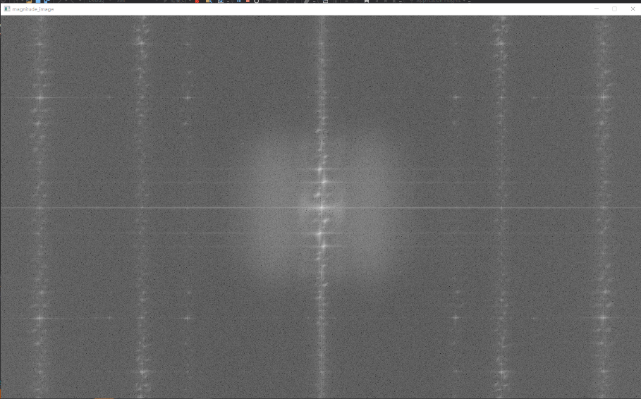
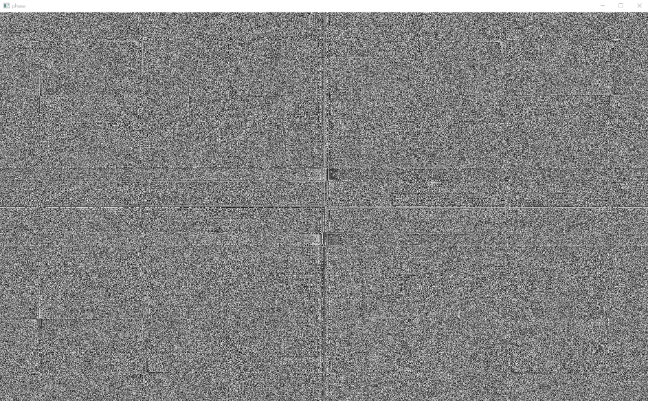
### c.呈現還原後圖像

對頻域圖像執行逆傅里葉變換，將圖像轉換回空域圖像。使用 magnitude() 函數計算逆變換後的複數矩陣的幅值，以獲取還原後圖像的強度信息。

## 程式碼

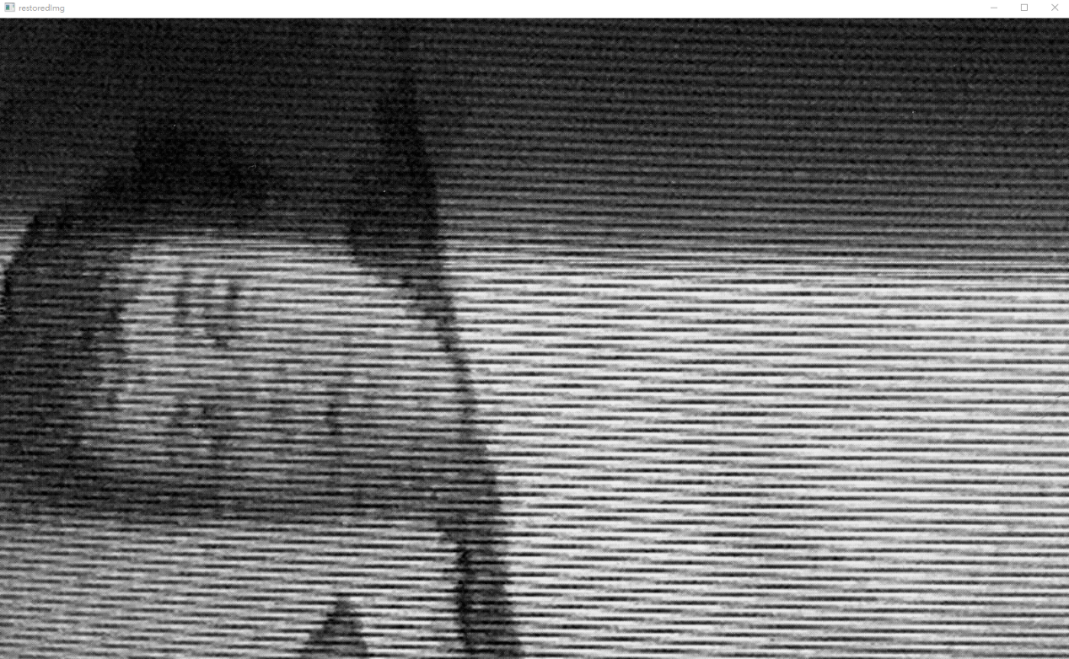


## 成果



|  |
| --- |
|  |
|  |

**頻譜圖 相位圖**

  
**還原後圖像**

影像還原練習

## 方法

### a.用notch filter去除雜訊

在作業3時做的圖象經傅立葉轉換後，依照頻譜圖上較明顯的亮點(某種頻域雜訊的位置)製作一個notch filter，套用在經dft後的transform\_image上(直接在transform\_image上畫上黑點，而畫的位置看頻譜圖上的亮點)

 畫上黑點方式:用滑鼠事件

當在某視窗有滑鼠事件發生時呼叫on\_mouse函式，在on\_mouse裡用得到的x、y座標來畫圖

用wiatKey()等待，在用滑鼠畫完後，按下enter鍵就可以產生圖

### b.逆傅立葉轉換

接下來再接續做逆傅立葉轉換，照作業3步驟將圖轉化回來

## 程式碼



## 成果



膚色偵測 Skin Color Detection

## 方法

將圖片轉換為 HSV 色彩空間，用cvtColor()將圖片轉換為 HSV 色彩空間。

定皮膚區域，設定hsv上下限，決定為h\_min = 4, h\_max = 19, s\_min = 43, s\_max = 231, v\_min = 35, v\_max = 255。

製作mask，inRange在原圖上找到顏色範圍，製作成mask。

臉部上色，圖中mask的範圍塗上紅色。

## 程式碼



## 成果

Run-Length Based Image Compression 練習

## 方法

**壓縮檔之格式**

儲存成dat檔，內有原圖長寬、通道順序、像素值與連續次數，儲存像素值的通道順序為R>G>B。

內容:

寬\*高

R G B

像素值 連續次數

像素值 連續次數

像素值 連續次數

...

...

解壓縮時，先讀出原圖片長寬，就能照通道順序與連續次數填入像素值，還原出原圖。

**輸入檔案方式:在main裡填入檔案路徑**

string inputfilename[3] = { "img1.bmp" ,"img2.bmp" ,"img3.bmp" };

**輸出為compressed\_+原檔名稱 的dat檔**

### a.壓縮方式

將圖片分成3個通道(RGB)，再把通道拉平成一維比較好做。

對每一個通道做壓縮，用flattenedRedChannel.at<uchar>看每pixel的值，如果一樣cnt就+1，直到不一樣，再把資料存進vector裡。

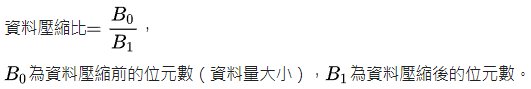
把壓縮後的資料序列放進vector容器，容器中分別放3個通道(R、G、B)的壓縮資料，每個壓縮資料以pair<uchar, int>存。

|  |
| --- |
|  |
|  |

### b.儲存資料

要存進dat檔裡，可以先把完整格式存成string，再直接輸出成dat檔案(二元檔)。

### c.算壓縮率



在c++中可以用Seek+tellg來算檔案byte數。

平均壓縮率 = 全部檔案的壓縮率加總 / 檔案次數

## 程式碼



## 成果

檔案: img1.bmp 壓縮率: 2.46887

檔案: img2.bmp 壓縮率: 1.43345

檔案: img3.bmp 壓縮率: 2.69067

平均壓縮率: 2.19766

