影像處理 Assignment4 - Hough Transform

姓名：吳嘉偉　學號：5105056013　日期：2018/01/13

1 灰階處理

1.1 把原始彩色影像轉成灰階影像

先把彩色的影像轉成灰階影像

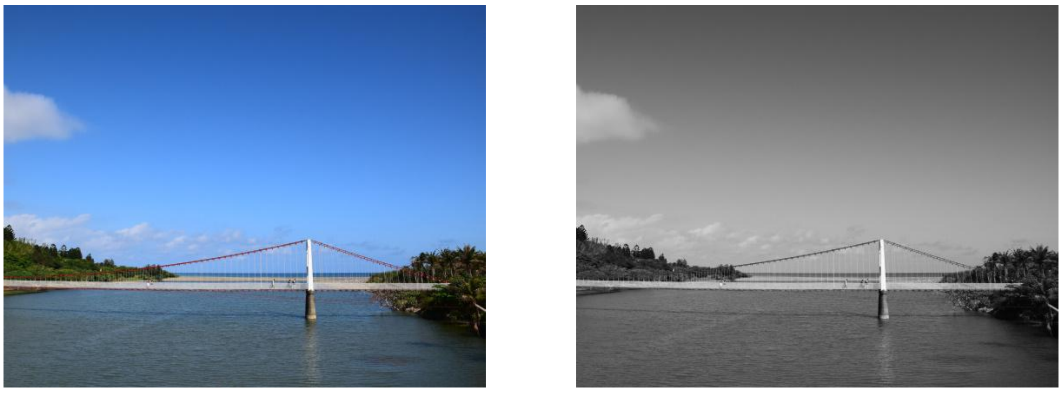

Figure 1: Source and Gray Image

1.2 程式碼

```
# 取得並儲存灰階照片
def readGrayImage(name):
    grayImage = cv2.imread(name, cv2.IMREAD_GRAYSCALE)
    savePhoto('gray_image', grayImage)
    return grayImage
```

2 利用 Hough Transform 找最長直線

2.1 Canny Edge

用 Canny Edge 找影像的邊緣

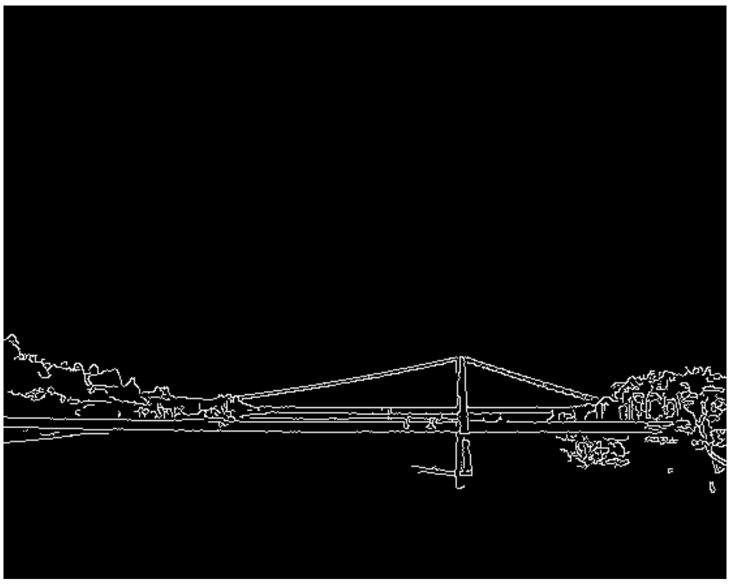

Figure 2: Canny Edge Image

程式碼

```
# 使用Canny Edge找出邊緣
def getCannyEdge(image):
    img = cv2.GaussianBlur(image, (3, 3), 0)
    cannyEdgeImage = cv2.Canny(img, 50, 150, apertureSize = 3)
    savePhoto('canny_image', cannyEdgeImage)
    return cannyEdgeImage
```

2.2 Hough Transform

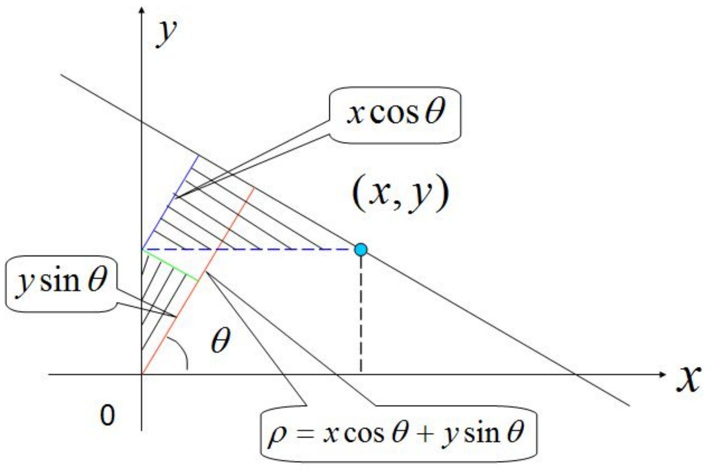

Figure 3: Hough Transform Theory

利用 $xcos\theta + ysin\theta = \rho$
把原本是 xy 點的座標軸轉換成原點到直線的垂直距離 (ρ) 以及與 x 軸夾角 (θ) 的座標

程式碼

```
# 取得Hough Transform圖片
def getHoughTransformImage(image, partNumber = 1000):
    height = image.shape[0]
    width = image.shape[1]
    minDistance = 0
    maxDistance = 0
    output_pixels = []

    for y in range(height):
        output_pixels.append([])
```

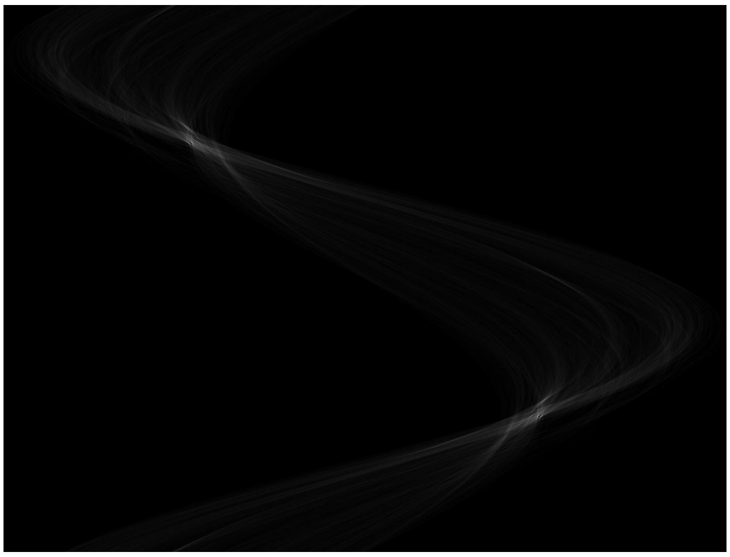

Figure 4: Hough Transform Image

```
    for x in range(width):
        if image[y][x] != 0:
            lineDistance = []
            for count in range(partNumber):
                angle = math.pi * (2 * count / partNumber - 1)
                distance = houghTransform(x, y, angle)
                lineDistance.append(distance)
                if distance < minDistance:
                    minDistance = distance
                if distance > maxDistance:
                    maxDistance = distance
            output_pixels[y].append(lineDistance)
        else:
            output_pixels[y].append([])

height = round(maxDistance - minDistance)
transData = np.zeros((height, partNumber), np.uint8)
maxCount = 0

for i in range(len(output_pixels)):
    row = output_pixels[i]
    for j in range(len(row)):
        line = row[j]
        for k in range(len(line)):
            p = int(line[k] - minDistance)
```

```
                transData[p][k] += 1
                if (transData[p][k] > maxCount):
                    maxCount = transData[p][k]

newImage = np.zeros((height, partNumber, 3), np.uint8)

for y in range(height):
    for x in range(partNumber):
        newImage[y][x] = int(transData[y][x] * 255 / maxCount)

savePhoto('hough_transform_image', newImage)
```

2.3 Longest Line

由於 Hough transform 把每一個像素 (x, y) 對應到距離與 x 軸夾角 (ρ, θ)，所以被對應到次數最高的一組 (ρ, θ)，就是影像上最長的一條直線

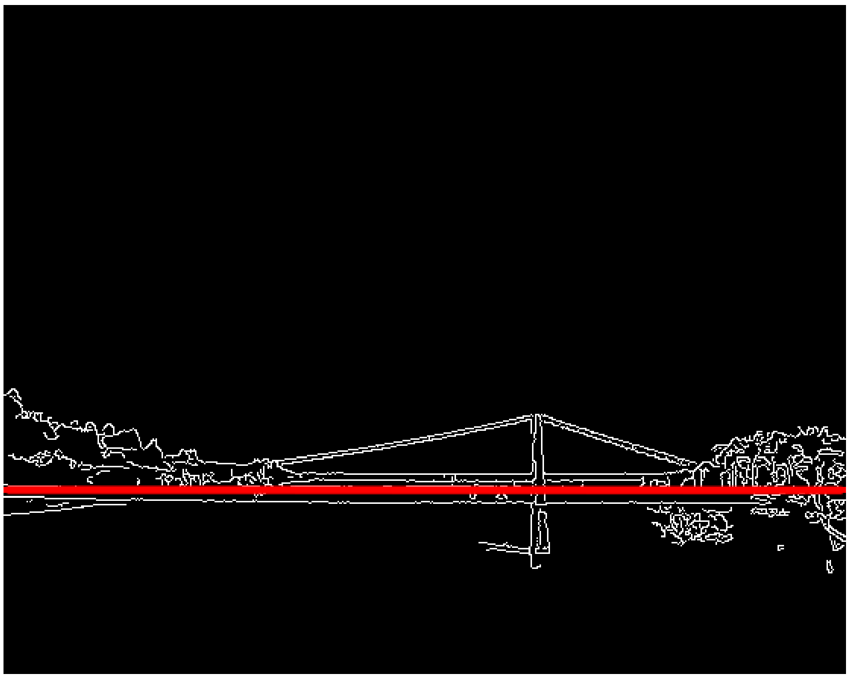

Figure 5: Longest Image

程式碼

```
# 畫出最長的直線
def drawLongestLine(cannyImage):
    cdst = cv2.cvtColor(cannyImage, cv2.COLOR_GRAY2BGR)
    lines = cv2.HoughLines(cannyImage, 1, np.pi / 180, 50, None, 50, 10)
    if lines is not None:
        for i in range(0, 1):
            rho = lines[i][0][0]
            theta = lines[i][0][1]
            a = math.cos(theta)
            b = math.sin(theta)
            x0 = a * rho
            y0 = b * rho
            pt1 = (int(x0 + 1000 * (-b)), int(y0 + 1000 * (a)))
            pt2 = (int(x0 - 1000 * (-b)), int(y0 - 1000 * (a)))
            cv2.line(cdst, pt1, pt2, (0, 0, 255), 3, cv2.LINE_AA)
    savePhoto('finalImage', cdst)
```