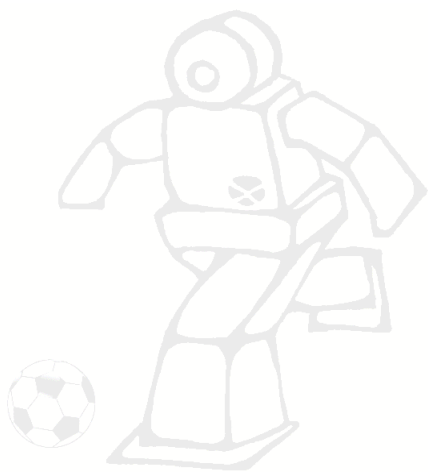


視覺感測技術應用實務

直方圖匹配 第二組

組員:李宗晏 608470042(100%)

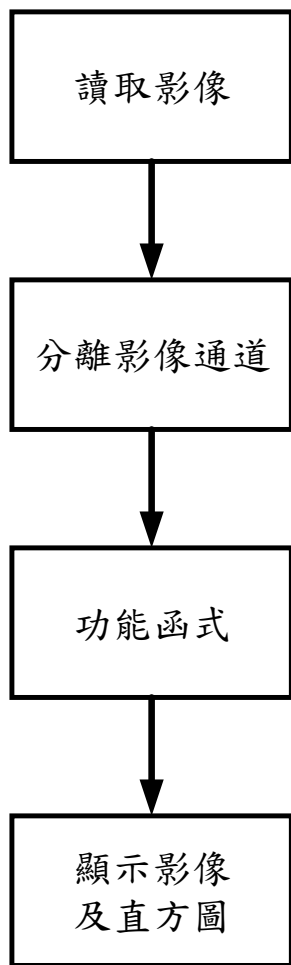


目錄

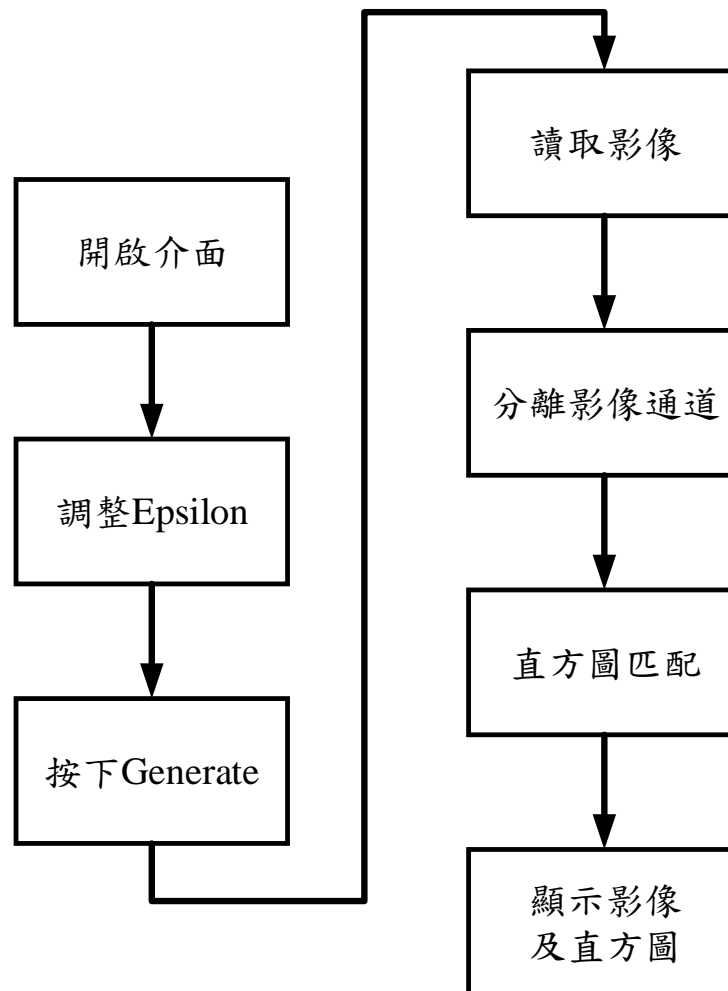
- 實現方法
- 獨特設計之處
- 結果展示
- 組員分工表

實現方法--動作流程圖

□ 一般函式流程



□ 直方圖匹配流程



實現方法--使用類別方法的類型

- 本次程式BaseIP和AlphaBlend的成員函式都使用static method，因為本次類別的成員函式不需要像是self或是cls等實例或類別的參考，所以使用static method可以比較簡明且有效率。
 - 簡明的部分在於不需要多接收一個無關緊要的引數
 - 效率在於一般的實例方法是bound method(是個object)且是在我們使用他的時候才生成，這會多花費一點的時間，而靜態方法並不會

實現方法--程式(cv2IP.py)

□ BaseIP類別

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

class HistIP(BaseIP):

    def __init__(self):
        super().__init__()

    class ColorType(enum.IntEnum):
        USE_RGB = 1
        USE_HSV = 2
        USE_YUV = 3

    @staticmethod
    def ImBGR2Gray(SrcImg):
        DstImg = cv2.cvtColor(SrcImg, cv2.COLOR_BGR2GRAY)
        return DstImg

    @staticmethod
    def ImBGRA2BGR(SrcImg):
        DstImg = np.array(SrcImg[:, :, :3])
        return DstImg
```

實現方法--程式(cv2IP.py)

□ 灰階直方圖

```
@staticmethod
def CalcGrayHist(SrcGray):
    GrayHist = cv2.calcHist([SrcGray], [0], None, [256], [0, 256])
    return GrayHist

@staticmethod
def ShowGrayHist(winname, GrayHist):
    plt.plot(GrayHist, "gray")
    plt.legend(["Gray"], loc="upper right")
    plt.title(winname)
    plt.xlabel("Bins")
    plt.ylabel("Number of pixels")
    plt.show()
```

實現方法--程式(cv2IP.py)

□ 色彩直方圖

```
@staticmethod
def CalcColorHist(SrcColor):
    BlueHist = cv2.calcHist([SrcColor], [0], None, [256], [0, 256])
    GreenHist = cv2.calcHist([SrcColor], [1], None, [256], [0, 256])
    RedHist = cv2.calcHist([SrcColor], [2], None, [256], [0, 256])
    ColorHist = cv2.merge([BlueHist, GreenHist, RedHist])
    return ColorHist

@staticmethod
def ShowColorHist(winname, ColorHist):
    plt.plot(ColorHist[:, :, 0], "b")
    plt.plot(ColorHist[:, :, 1], "g")
    plt.plot(ColorHist[:, :, 2], "r")
    plt.legend(["Blue", "Green", "Red"], loc="upper right")
    plt.title(winname)
    plt.xlabel("Bins")
    plt.ylabel("Number of pixels")
    plt.show()
```

實現方法--程式(cv2IP.py)

□ 影像直方圖等化

```
@staticmethod
def MonoEqualize(SrcGray):
    EqualizedGray = cv2.equalizeHist(SrcGray)
    return EqualizedGray

@staticmethod
def ColorEqualize(SrcColor, CType = ColorType.USE_HSV):
    if CType == HistIP.ColorType.USE_RGB:
        EqualizedBlue = cv2.equalizeHist(SrcColor[:, :, 0])
        EqualizedGreen = cv2.equalizeHist(SrcColor[:, :, 1])
        EqualizedRed = cv2.equalizeHist(SrcColor[:, :, 2])
        EqualizedColor = cv2.merge([EqualizedBlue, EqualizedGreen, EqualizedRed])
    elif CType == HistIP.ColorType.USE_HSV:
        SrcHSV = cv2.cvtColor(SrcColor, cv2.COLOR_BGR2HSV)
        EqualizedHSV = np.array(SrcHSV)
        EqualizedHSV[:, :, 2] = cv2.equalizeHist(SrcHSV[:, :, 2])
        EqualizedColor = cv2.cvtColor(EqualizedHSV, cv2.COLOR_HSV2BGR)
    elif CType == HistIP.ColorType.USE_YUV:
        SrcYUV = cv2.cvtColor(SrcColor, cv2.COLOR_BGR2YUV)
        EqualizedYUV = np.array(SrcYUV)
        EqualizedYUV[:, :, 0] = cv2.equalizeHist(SrcYUV[:, :, 0])
        EqualizedColor = cv2.cvtColor(EqualizedYUV, cv2.COLOR_YUV2BGR)
    return EqualizedColor
```


實現方法--程式(cv2IP.py)

□ PDF、CDF的直方圖

```
@staticmethod
def CalPDFGrayHist(SrcImg):
    GrayHist = HistIP.CalcGrayHist(SrcImg)
    PDFGrayHist = GrayHist / SrcImg.size
    return PDFGrayHist

@staticmethod
def CalPDFColorHist(SrcImg):
    ColorHist = HistIP.CalcColorHist(SrcImg)
    PDFColorHist = ColorHist / SrcImg[:, :, 0].size
    return PDFColorHist

@staticmethod
def CalCDFGrayHist(SrcImg):
    PDFGrayHist = HistIP.CalPDFGrayHist(SrcImg)
    CDFGrayHist = np.zeros(PDFGrayHist.shape)
    CDFGrayHist[0, :] = PDFGrayHist[0, :]
    for i in range(1, 256, 1):
        CDFGrayHist[i, :] = CDFGrayHist[i-1, :] + PDFGrayHist[i, :]
    return CDFGrayHist

@staticmethod
def CalCDFColorHist(SrcImg):
    PDFColorHist = HistIP.CalPDFColorHist(SrcImg)
    CDFColorHist = np.zeros(PDFColorHist.shape)
    CDFColorHist[0, :, :] = PDFColorHist[0, :, :]
    for i in range(1, 256, 1):
        CDFColorHist[i, :, :] = CDFColorHist[i-1, :, :] + PDFColorHist[i, :, :]
    return CDFColorHist
```

實現方法--程式(cv2IP.py)

□ LUT

```
@staticmethod
def CalculateLUT(SrcCDFHist, RefCDFHist, Epsilon = 0.05):
    LUT = np.zeros(256, dtype=np.int)
    Last = 0
    for i in range(256):
        for j in range(Last, 256, 1):
            if abs(RefCDFHist[j,0] - SrcCDFHist[i,0]) < Epsilon or RefCDFHist[j,0] > SrcCDFHist[i,0]:
                LUT[i] = j
                Last = j
                break
    return LUT
```

實現方法--程式(cv2IP.py)

□ RGB 色彩空間的直方圖匹配

```
@staticmethod
def HistMatching(SrcImg, RefImg, Epsilon, CType = ColorType.USE_HSV):
    if CType == HistIP.ColorType.USE_RGB:
        #-----CDF-----#
        Src_CDFHist = HistIP.CalcCDFColorHist(SrcImg)
        Ref_CDFHist = HistIP.CalcCDFColorHist(RefImg)
        #-----LUT-----#
        BlueLUT = HistIP.CalculateLUT(Src_CDFHist[:, :, 0], Ref_CDFHist[:, :, 0], Epsilon)
        GreenLUT = HistIP.CalculateLUT(Src_CDFHist[:, :, 1], Ref_CDFHist[:, :, 1], Epsilon)
        RedLUT = HistIP.CalculateLUT(Src_CDFHist[:, :, 2], Ref_CDFHist[:, :, 2], Epsilon)
        LUT = cv2.merge([BlueLUT, GreenLUT, RedLUT])
        DstImg = np.array(SrcImg)
        for i in range(3):
            DstImg[:, :, i] = cv2.LUT(SrcImg[:, :, i], LUT[:, 0, i])
```

實現方法--程式(cv2IP.py)

□ HSV 色彩空間的直方圖匹配

```
elif CType == HistIP.ColorType.USE_HSV:
    SrcHSV = cv2.cvtColor(SrcImg, cv2.COLOR_BGR2HSV)
    RefHSV = cv2.cvtColor(RefImg, cv2.COLOR_BGR2HSV)
    #-----CDF-----#
    Src_CDFHist = HistIP.CalcCDFGrayHist(SrcHSV[:, :, 2])
    Ref_CDFHist = HistIP.CalcCDFGrayHist(RefHSV[:, :, 2])
    #-----LUT-----#
    LUT = HistIP.CalculateLUT(Src_CDFHist, Ref_CDFHist, Epsilon)
    DstHSV = np.array(SrcHSV)
    DstHSV[:, :, 2] = cv2.LUT(SrcHSV[:, :, 2], LUT)
    DstImg = cv2.cvtColor(DstHSV, cv2.COLOR_HSV2BGR)
```

實現方法--程式(cv2IP.py)

□ YUV 色彩空間的直方圖匹配

```
elif CType == HistIP.ColorType.USE_YUV:
    SrcYUV = cv2.cvtColor(SrcImg, cv2.COLOR_BGR2YUV)
    RefYUV = cv2.cvtColor(RefImg, cv2.COLOR_BGR2YUV)
    #-----CDF-----#
    Src_CDFHist = HistIP.CalcCDFGrayHist(SrcYUV[:, :, 0])
    Ref_CDFHist = HistIP.CalcCDFGrayHist(RefYUV[:, :, 0])
    #-----LUT-----#
    LUT = HistIP.CalculateLUT(Src_CDFHist, Ref_CDFHist, Epsilon)
    DstYUV = np.array(SrcYUV)
    DstYUV[:, :, 0] = cv2.LUT(SrcYUV[:, :, 0], LUT, Epsilon)
    DstImg = cv2.cvtColor(DstYUV, cv2.COLOR_YUV2BGR)
return DstImg
```

實現方法--程式(Project2.py)

□ 主函式

```
import cv2
import numpy as np
import cv2IP
import tkinter as tk
```

□ 創建HistIP的物件

```
def MyColorHistMatching(Epsilon):
    IP = cv2IP.HistIP()
```

- 使用物件導向方式來呼叫類別中的成員函式

實現方法--程式(Project2.py)

□ 灰階影像

```
def MyShowGrayImage():  
    IP = cv2IP.HistIP()  
    SrcImg = IP.ImRead("img/foreGroundAsset.png")  
    if SrcImg.shape[2] == 4:  
        F_BGR = IP.ImBGRA2BGR(SrcImg)  
    else:  
        F_BGR = np.array(SrcImg)  
    F_Gray = IP.ImBGR2Gray(F_BGR)  
    IP.ImWindow("ForeGround Gray Image")  
    IP.ImShow("ForeGround Gray Image", F_Gray)  
    cv2.waitKey(0)  
    del IP
```

實現方法--程式(Project2.py)

□ 灰階直方圖

```
def MyShowGrayHistogram():  
    IP = cv2IP.HistIP()  
    SrcImg = IP.ImRead("img/foreGroundAsset.png")  
    if SrcImg.shape[2] == 4:  
        F_BGR = IP.ImBGRA2BGR(SrcImg)  
    else:  
        F_BGR = np.array(SrcImg)  
    F_Gray = IP.ImBGR2Gray(F_BGR)  
    F_Hist = IP.CalcGrayHist(F_Gray)  
    IP.ShowGrayHist("ForeGround Gray Hist", F_Hist)  
    del IP
```


實現方法--程式(Project2.py)

□ 色彩直方圖

```
def MyShowColorHistogram():  
    IP = cv2IP.HistIP()  
    SrcImg = IP.ImRead("img/foreGroundAsset.png")  
    if SrcImg.shape[2] == 4:  
        F_BGR = IP.ImBGRA2BGR(SrcImg)  
    else:  
        F_BGR = np.array(SrcImg)  
    F_Hist = IP.CalcColorHist(F_BGR)  
    IP.ShowColorHist("ForeGround Color Hist", F_Hist)  
    del IP
```

實現方法--程式(Project2.py)

□ 灰階直方圖等化

```
def MyMonoHistEqualize():
    IP = cv2IP.HistIP()
    SrcImg = IP.ImRead("img/InputIm_1_FixdPoint.bmp")
    if SrcImg.shape[2] == 4:
        F_BGR = IP.ImBGRA2BGR(SrcImg)
    else:
        F_BGR = np.array(SrcImg)
    F_Gray = IP.ImBGR2Gray(F_BGR)
    F_Eq = IP.MonoEqualize(F_Gray)
    F_GrayHist = IP.CalcGrayHist(F_Gray)
    F_EqualizedHist = IP.CalcGrayHist(F_Eq)
    IP.ShowGrayHist("Foreground Gray Hist", F_GrayHist)
    IP.ShowGrayHist("Foreground Equalized Hist", F_EqualizedHist)
    IP.ImWindow("Foreground Gray")
    IP.ImShow("Foreground Gray", F_Gray)
    IP.ImWindow("Foreground Gray Equalized")
    IP.ImShow("Foreground Gray Equalized", F_Eq)
    cv2.waitKey(0)
    del IP
```

實現方法--程式(Project2.py)

□ 色彩直方圖等化

```
def MyColorHistEqualize(CType):  
    IP = cv2IP.HistIP()  
    SrcImg = IP.ImRead("img/InputIm_1_FixdPoint.bmp")  
    if SrcImg.shape[2] == 4:  
        F_BGR = IP.ImBGRA2BGR(SrcImg)  
    else:  
        F_BGR = np.array(SrcImg)  
    F_Eq = IP.ColorEqualize(SrcImg, CType)  
    F_ColorHist = IP.CalcColorHist(SrcImg)  
    F_EqualizedHist = IP.CalcColorHist(F_Eq)  
    IP.ShowColorHist("ForeGround Color Hist", F_ColorHist)  
    IP.ShowColorHist("ForeGround Equalized Hist", F_EqualizedHist)  
    IP.ImWindow("ForeGround Color")  
    IP.ImShow("ForeGround Color", SrcImg)  
    IP.ImWindow("ForeGround Color Equalized")  
    IP.ImShow("ForeGround Color Equalized", F_Eq)  
    cv2.waitKey(0)  
    del IP
```

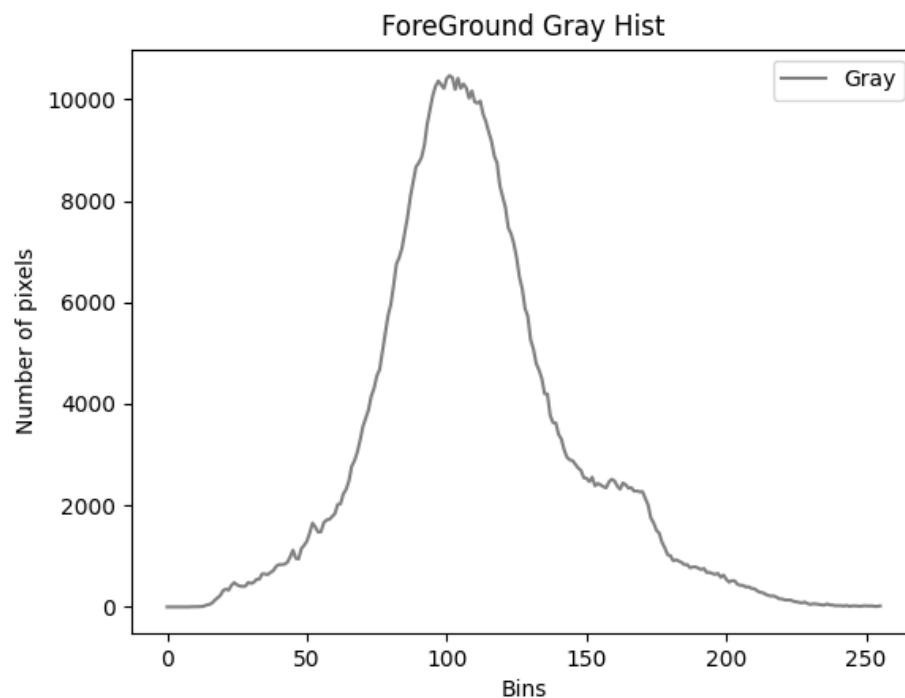
實現方法--程式(Project2.py)

□ 直方圖匹配

```
def MyColorHistMatching(Epsilon):
    IP = cv2IP.HistIP()
    SrcImg = IP.ImRead("img/swan.png")
    RefImg = IP.ImRead("img/InputIm_1_FixdPoint.bmp")
    if SrcImg.shape[2] == 4:
        Src_BGR = IP.ImBGRA2BGR(SrcImg)
    else:
        Src_BGR = np.array(SrcImg)
    if RefImg.shape[2] == 4:
        Ref_BGR = IP.ImBGRA2BGR(RefImg)
    else:
        Ref_BGR = np.array(RefImg)
    OutImg = IP.HistMatching(Src_BGR, Ref_BGR, Epsilon, IP.ColorType.USE_HSV)
    IP.ImShow("Original Image", SrcImg)
    IP.ImShow("Reference Image", RefImg)
    IP.ImShow("Processed Image", OutImg)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
    del IP
```

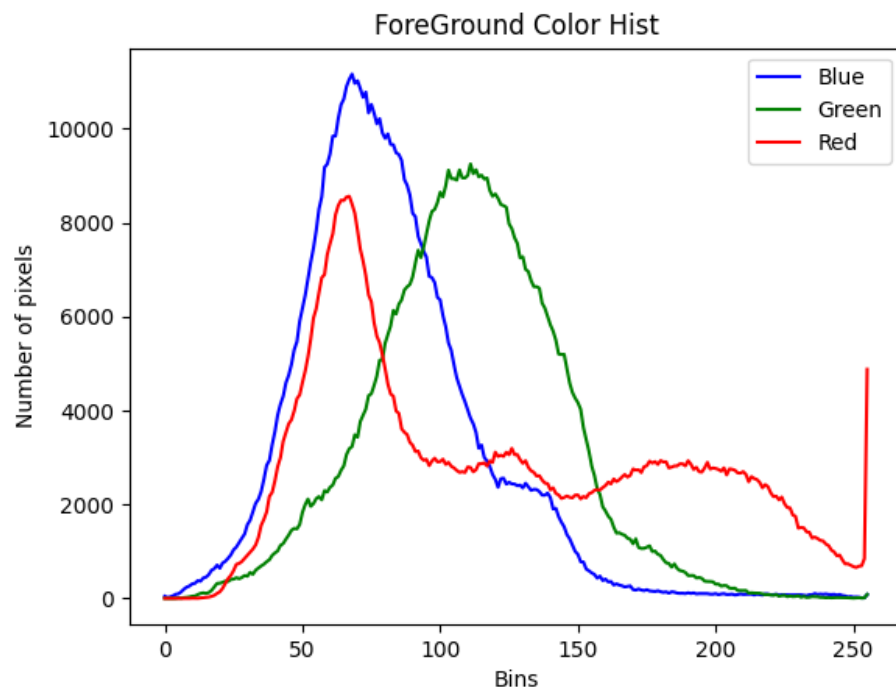
結果展示

灰階直方圖



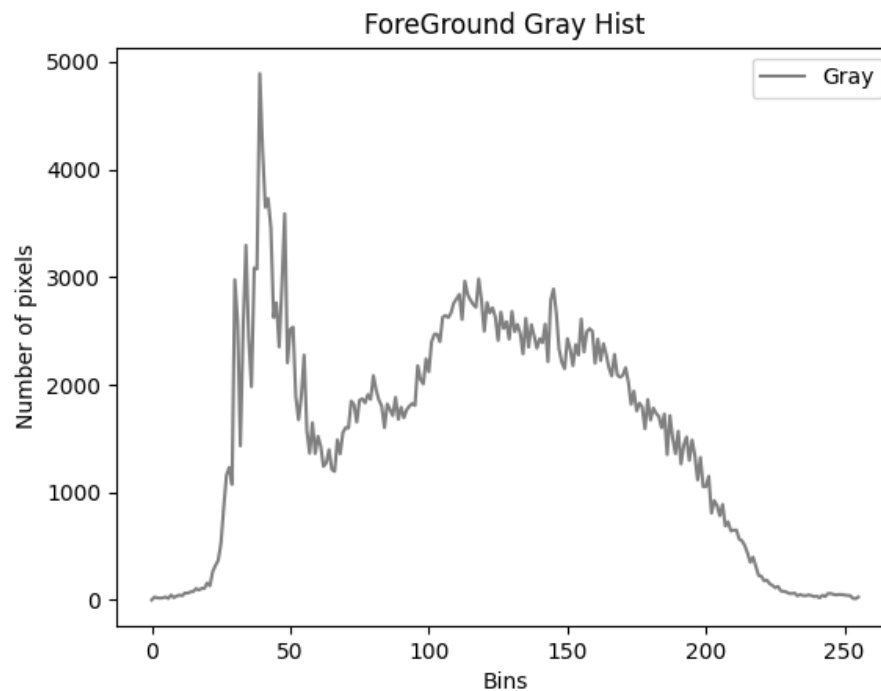
結果展示

色彩直方圖



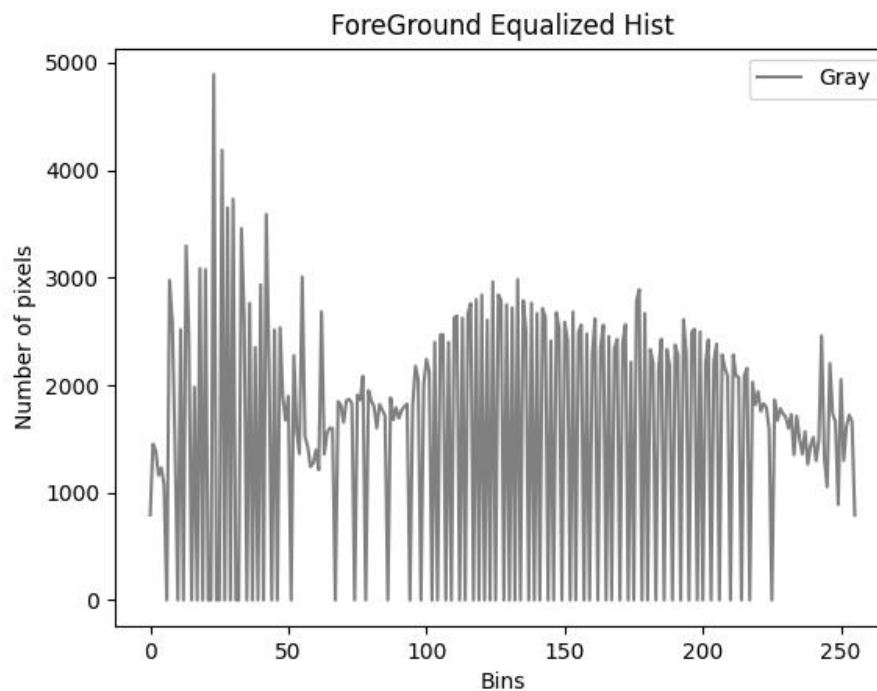
結果展示

灰階直方圖等化(來源圖)



結果展示

灰階直方圖等化(結果圖)



結果展示

□ 色彩直方圖等化(RGB 色彩空間)



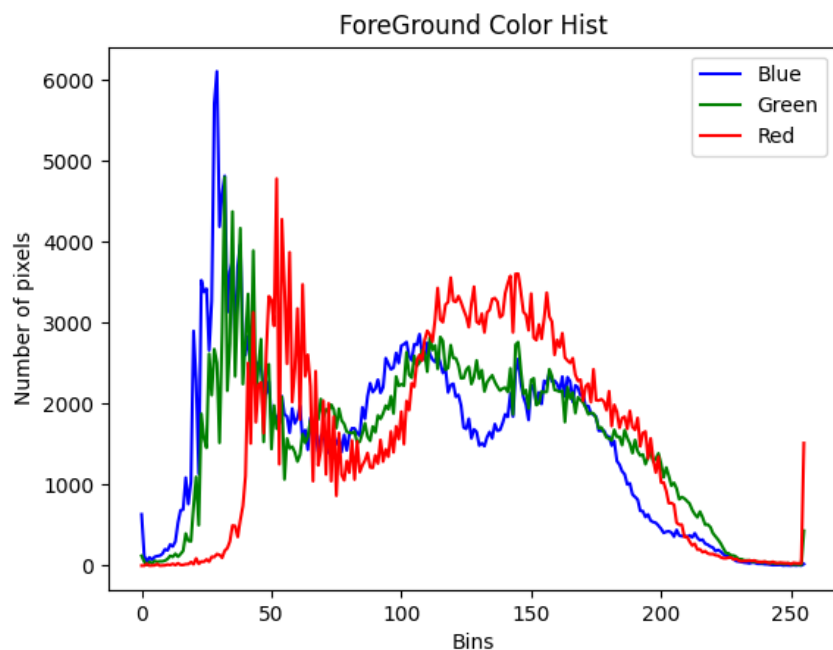
來源影像圖



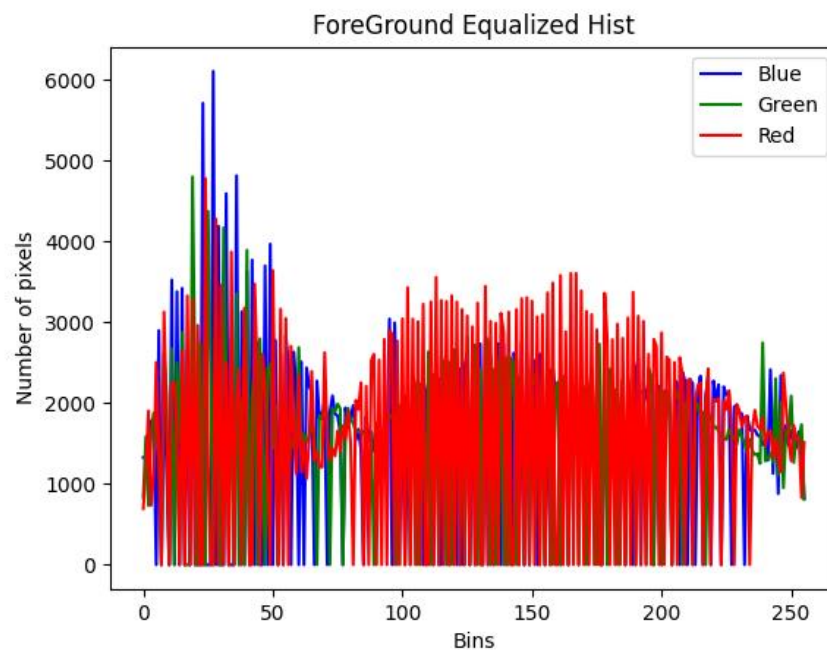
結果影像圖

結果展示

色彩直方圖等化(RGB色彩空間)



來源直方圖



結果直方圖

結果展示

□ 色彩直方圖等化(HSV色彩空間)



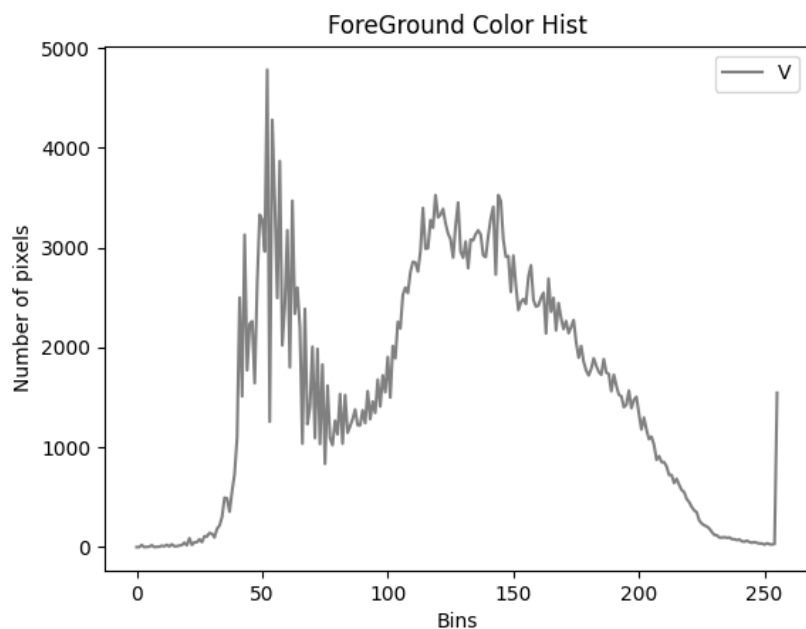
來源影像圖



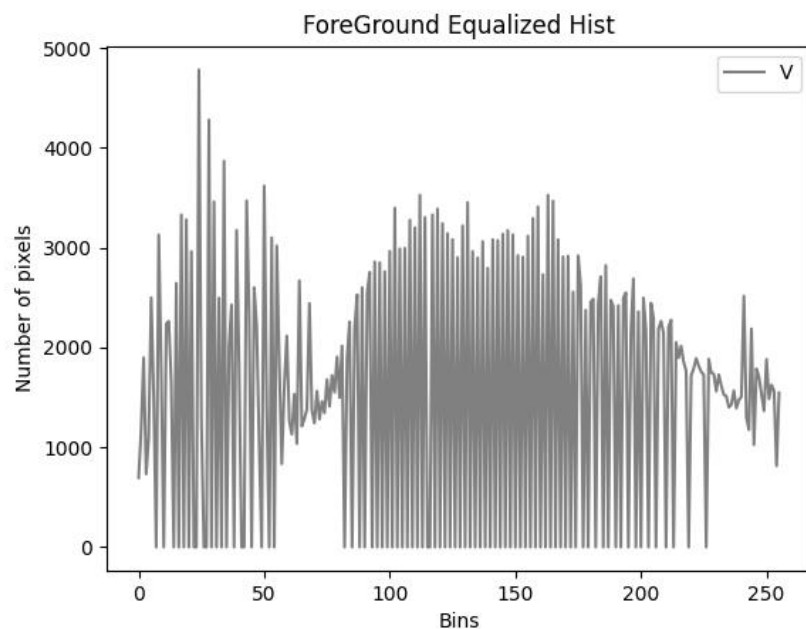
結果影像圖

結果展示

□ 色彩直方圖等化(HSV色彩空間)



來源直方圖



結果直方圖

結果展示

□ 色彩直方圖等化(YUV色彩空間)



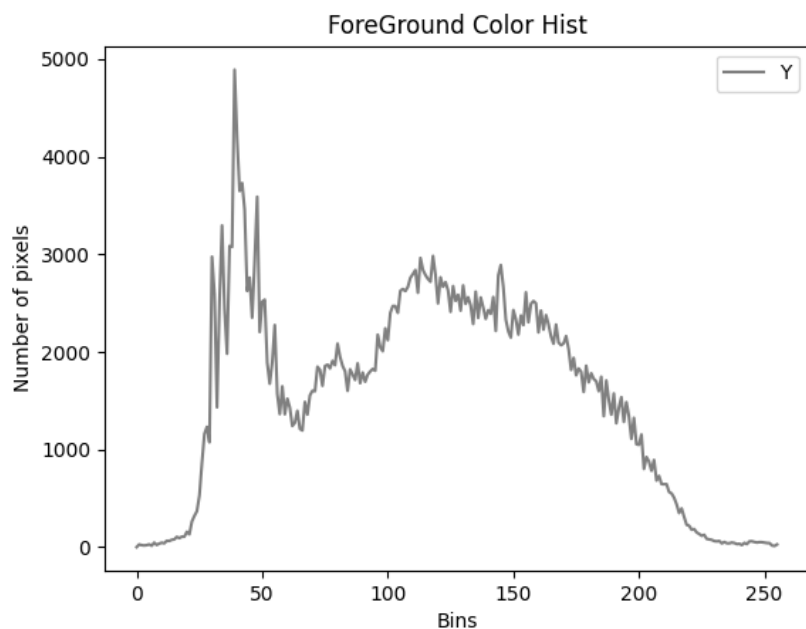
來源影像圖



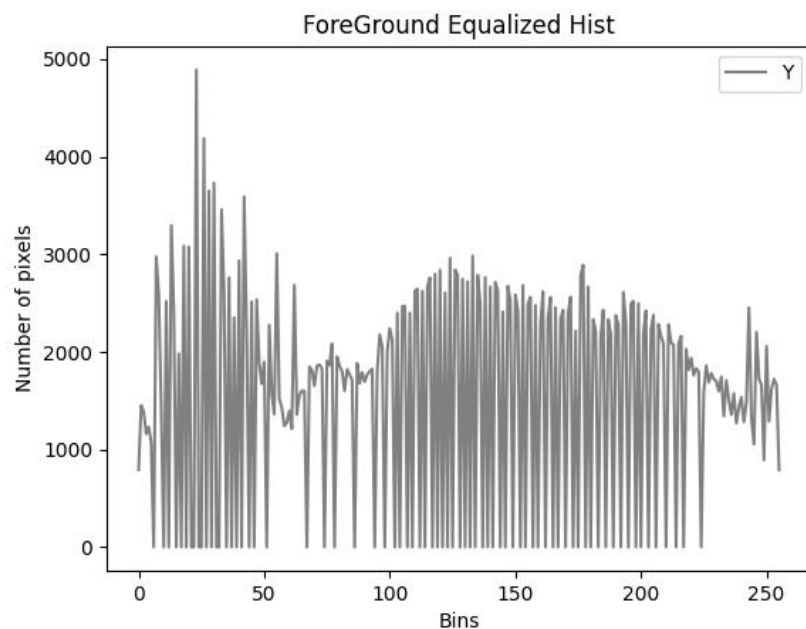
結果影像圖

結果展示

色彩直方圖等化(YUV色彩空間)



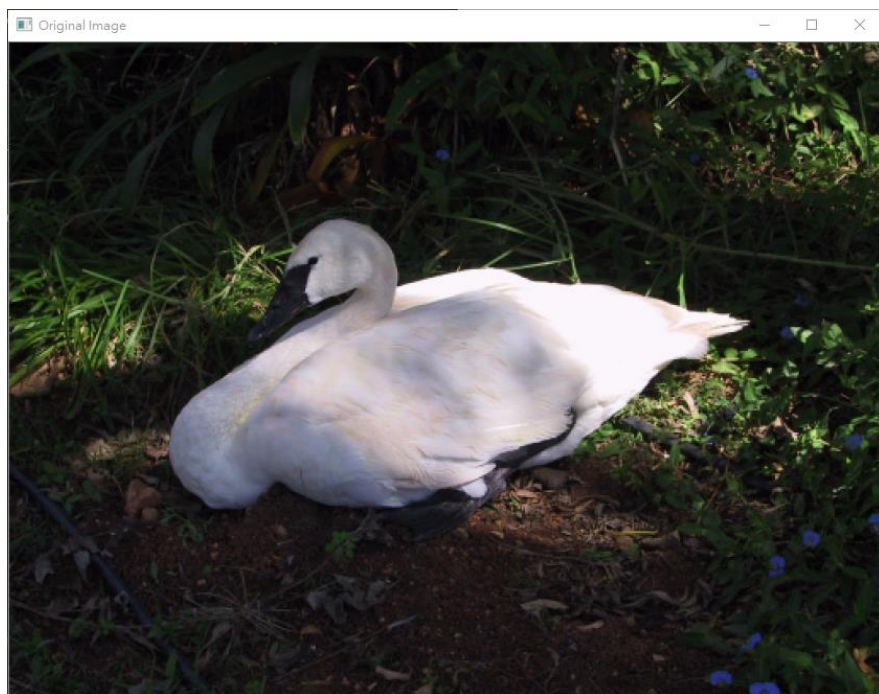
來源直方圖



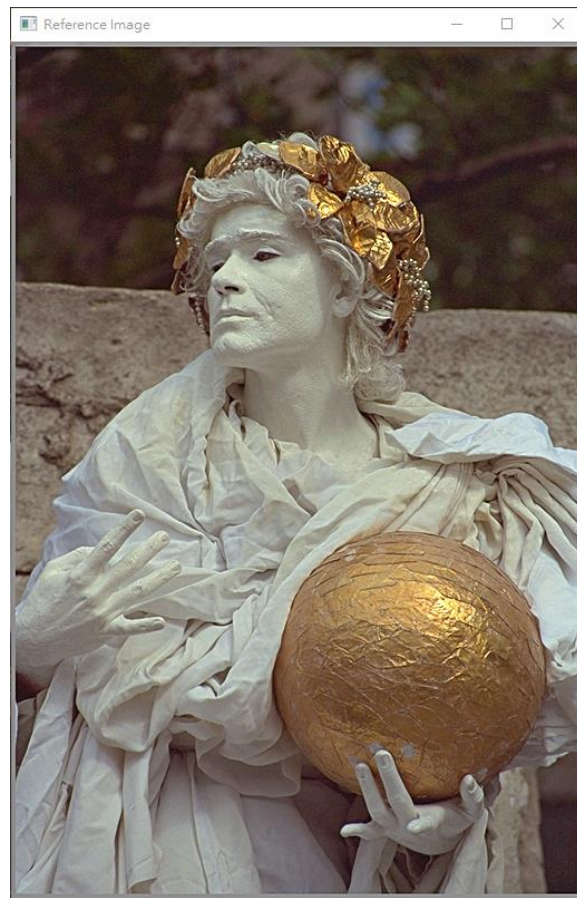
結果直方圖

結果展示

□ 色彩直方圖匹配



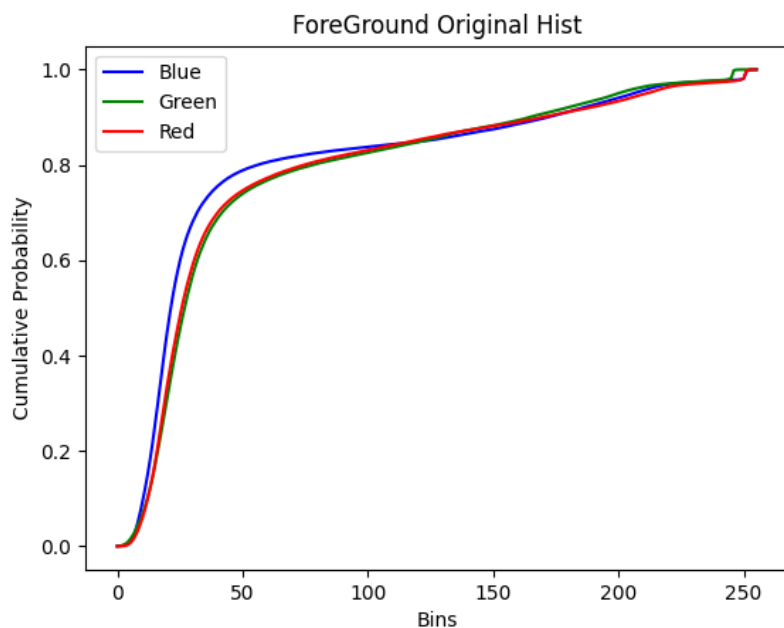
來源影像圖



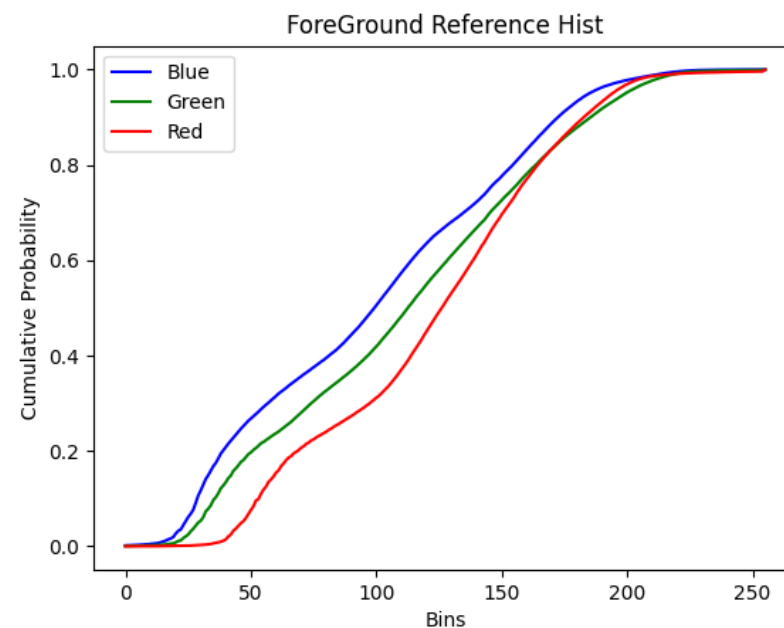
參考影像圖

結果展示

直方圖匹配(RGB色彩空間)



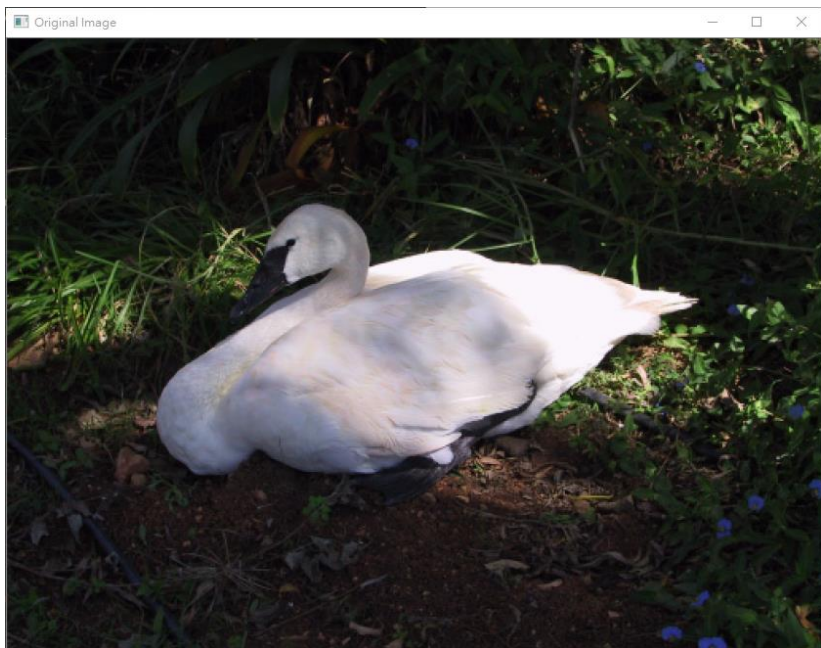
來源直方圖



參考直方圖

結果展示

□ 直方圖匹配(RGB色彩空間)



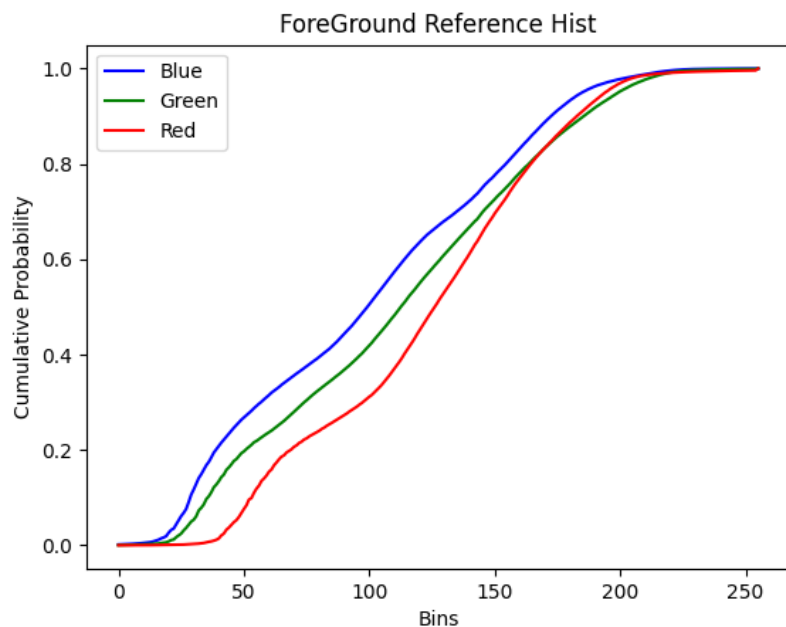
來源影像圖



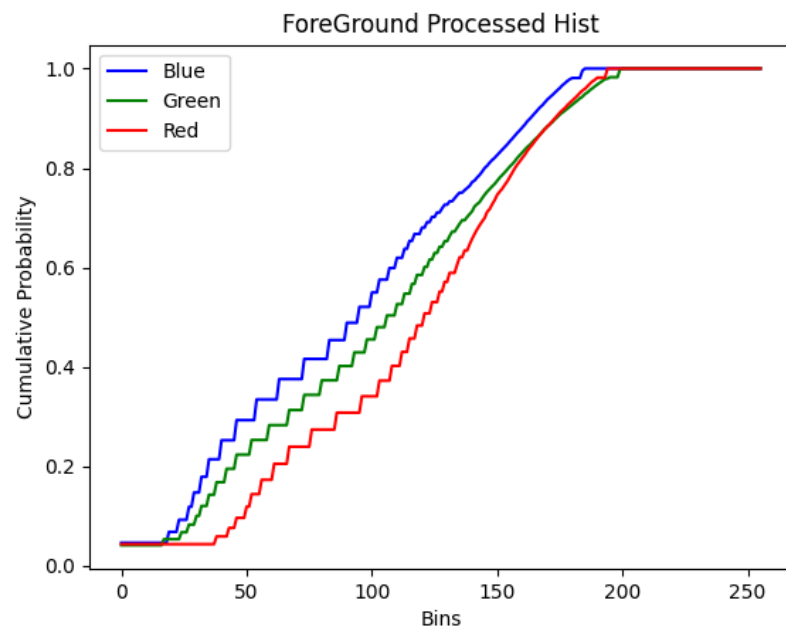
結果影像圖

結果展示

直方圖匹配(RGB色彩空間)



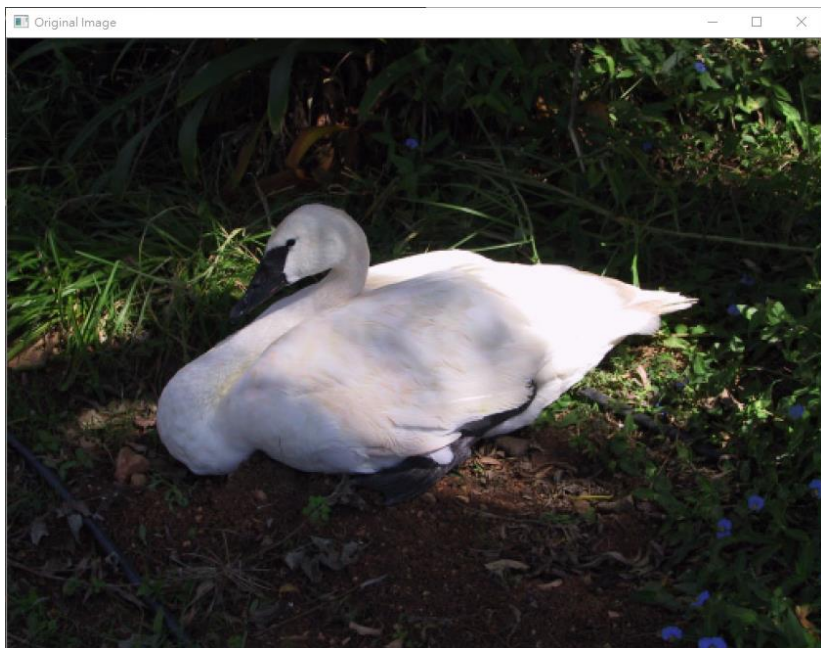
參考直方圖



結果直方圖

結果展示

□ 直方圖匹配(HSV色彩空間)



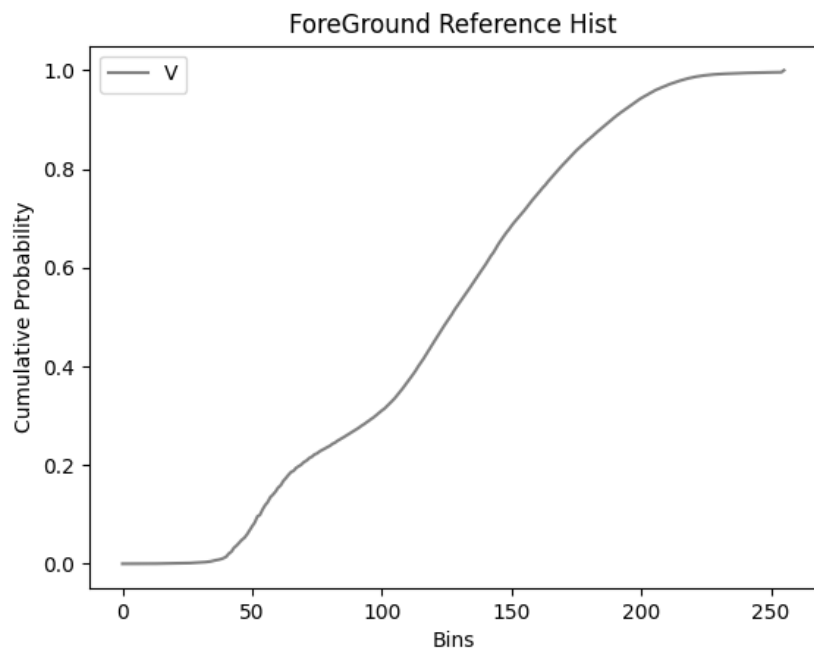
來源影像圖



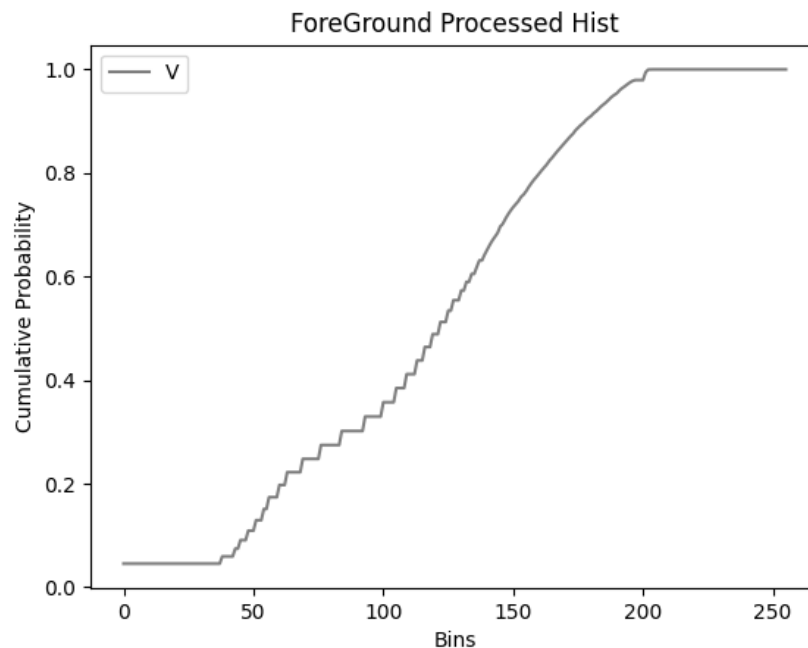
結果影像圖

結果展示

直方圖匹配(HSV色彩空間)



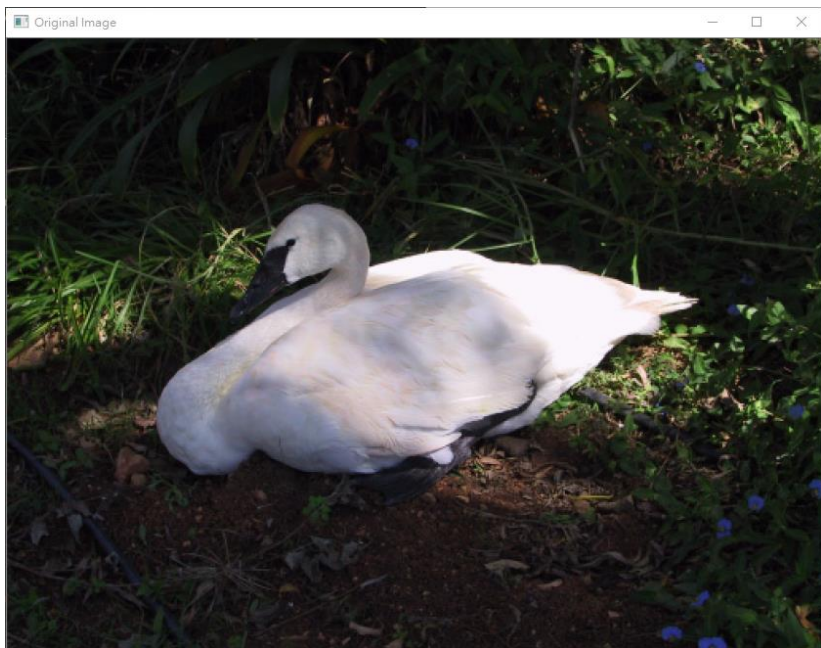
參考直方圖



結果直方圖

結果展示

□ 直方圖匹配(YUV色彩空間)



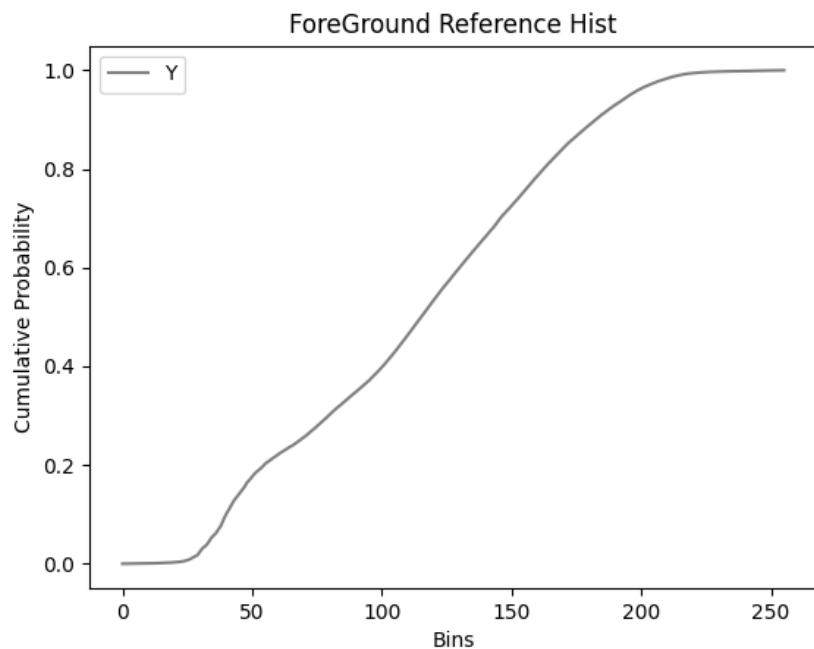
來源影像圖



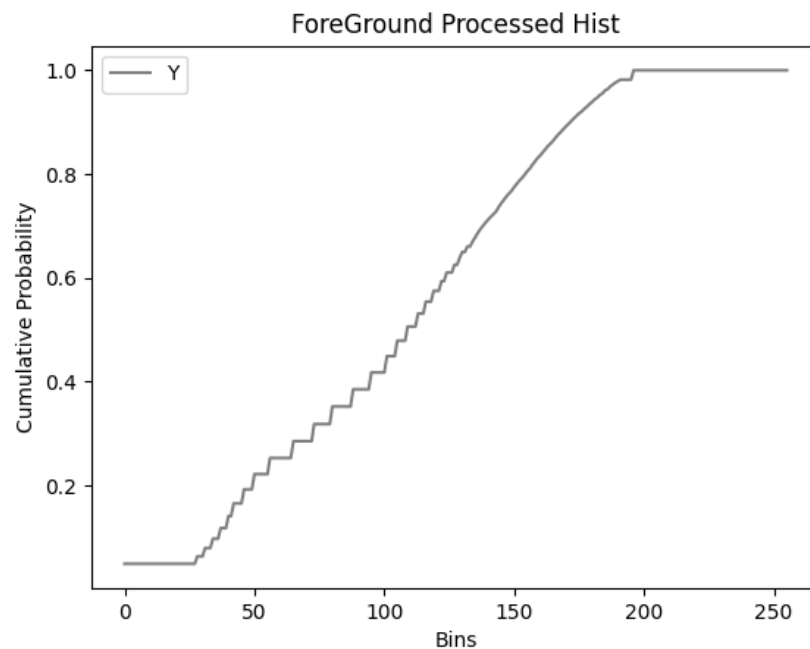
結果影像圖

結果展示

直方圖匹配(YUV色彩空間)



參考直方圖



結果直方圖

獨特設計之處

□ 主函式

```
if __name__ == '__main__':
    global GlobalEpsilon
    GlobalEpsilon = 0.0

    # 建立主視窗和 Frame (把元件變成群組的容器)
    window = tk.Tk()
    top_frame = tk.Frame(window)
    window.title('Hist Matching')
    window.geometry('320x140')
    window.configure(background='white')
    top_frame.pack()
    header_label = tk.Label(window, text='Hist Matching')
    header_label.pack()

    # 以下為 ChangeEpsilon_frame 群組
    ChangeEpsilon_frame = tk.Frame(window)
    ChangeEpsilon_frame.pack(side=tk.TOP)

    # 建立事件處理函式 (event handler) , 透過元件 command 參數存取
    def ChangeEpsilon(epsilon):
        header_label.config(text='Epsilon of Hist Matching is ' + epsilon)
        global GlobalEpsilon
        GlobalEpsilon = float(epsilon)

    # 將元件分為Scale 加入主視窗
    scale = tk.Scale(ChangeEpsilon_frame, label='epsilon', from_=0.0, to=0.1, orient=tk.HORIZONTAL, length=200, showvalue=1, tickinterval=2, resolution=0.01, command=ChangeEpsilon)
    scale.pack()
```

獨特設計之處

□ 主函式

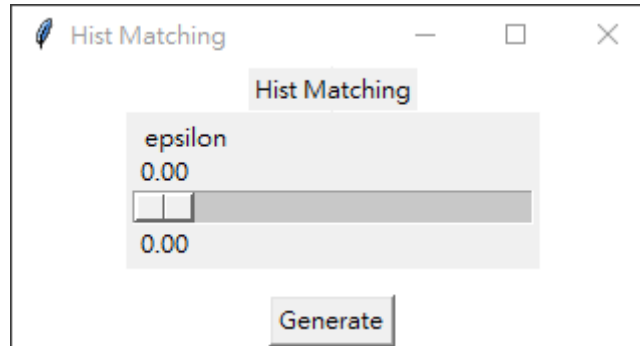
```
# 建立事件處理函式 (event handler)，透過元件 command 參數存取
def HistMatching():
    MyColorHistMatching(GlobalEpsilon)

# 將元件分為Scale 加入主視窗
bottom_frame = tk.Frame(window)
bottom_frame.pack(side=tk.BOTTOM)
# 以下為 bottom 群組
bottom_button = tk.Button(bottom_frame, text='Generate', fg='black', command=HistMatching)
# 讓系統自動擺放元件 (靠下方)
bottom_button.pack()

# 運行主程式
window.mainloop()
```

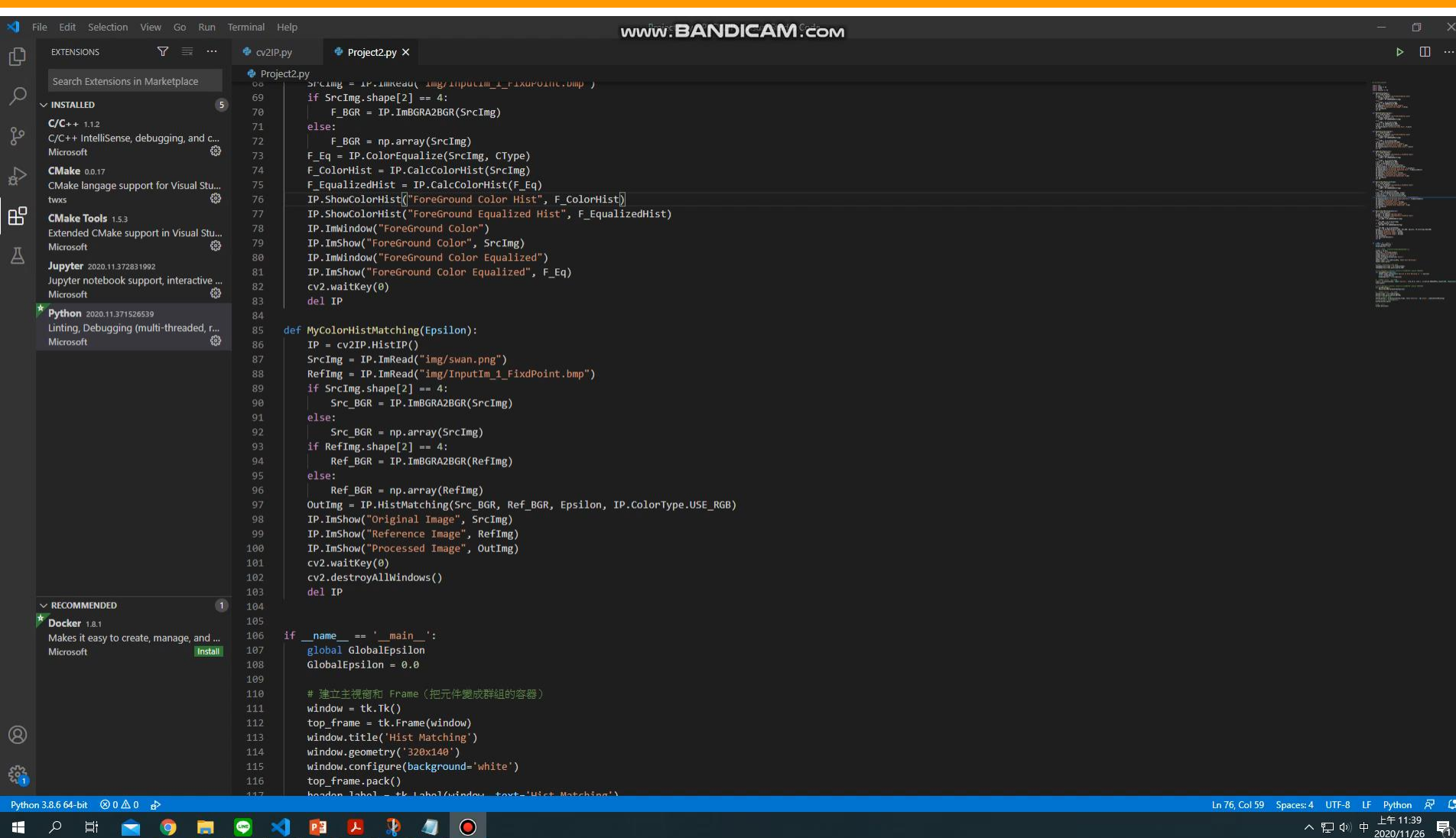

獨特設計之處

□ 調整Epsilon



```
@staticmethod
def CalculateLUT(SrcCDFHist, RefCDFHist, Epsilon = 0.05):
    LUT = np.zeros(256, dtype=np.int)
    Last = 0
    for i in range(256):
        for j in range(Last, 256, 1):
            if abs(RefCDFHist[j,0] - SrcCDFHist[i,0]) < Epsilon or RefCDFHist[j,0] > SrcCDFHist[i,0]:
                LUT[i] = j
                Last = j
                break
    return LUT
```

獨特設計之處(RGB 色彩空間)

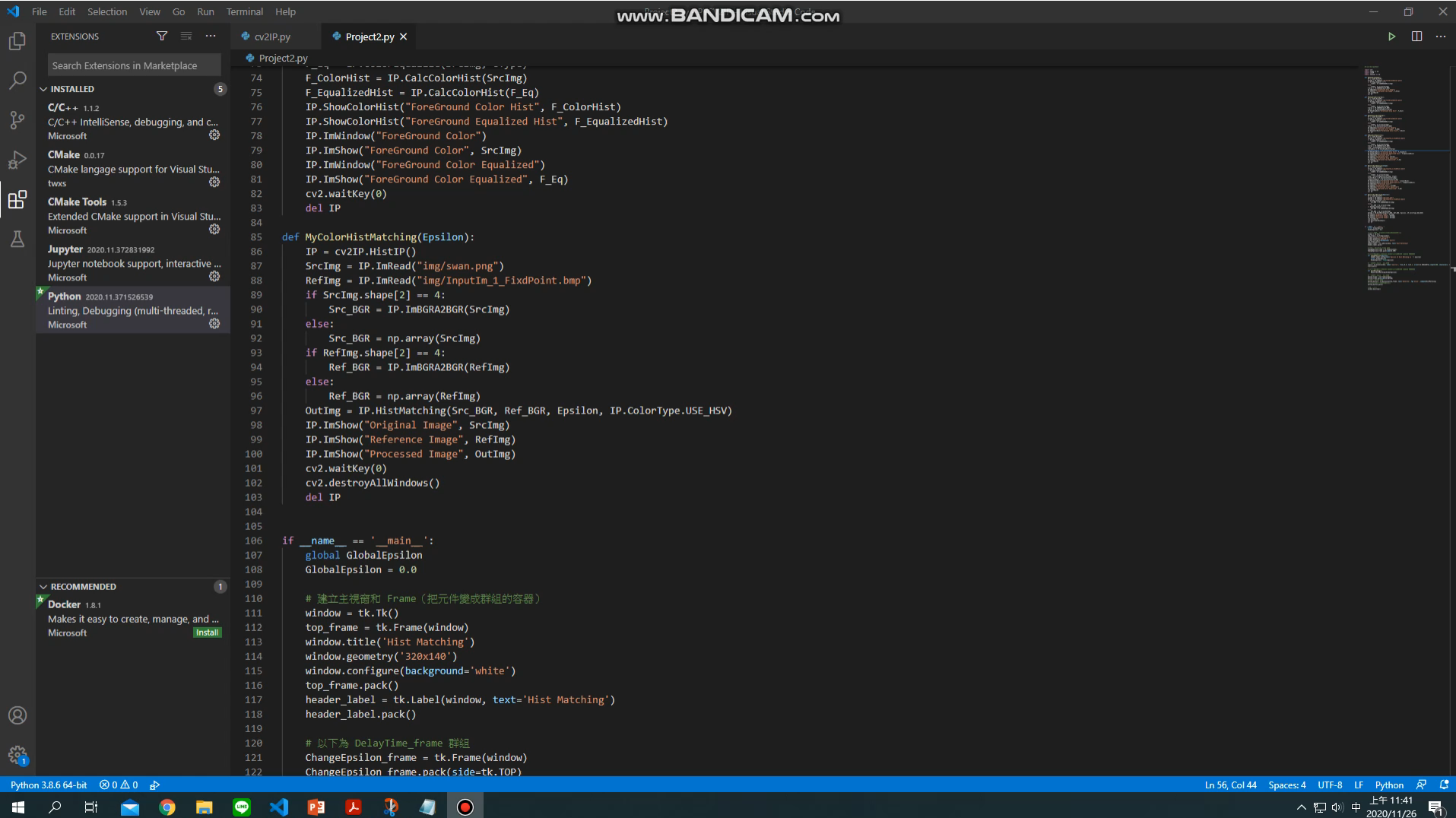


The screenshot shows a Python IDE with a dark theme. The left sidebar displays the 'EXTENSIONS' panel with a search bar and a list of installed and recommended extensions. The main editor window shows a file named 'Project2.py' with the following code:

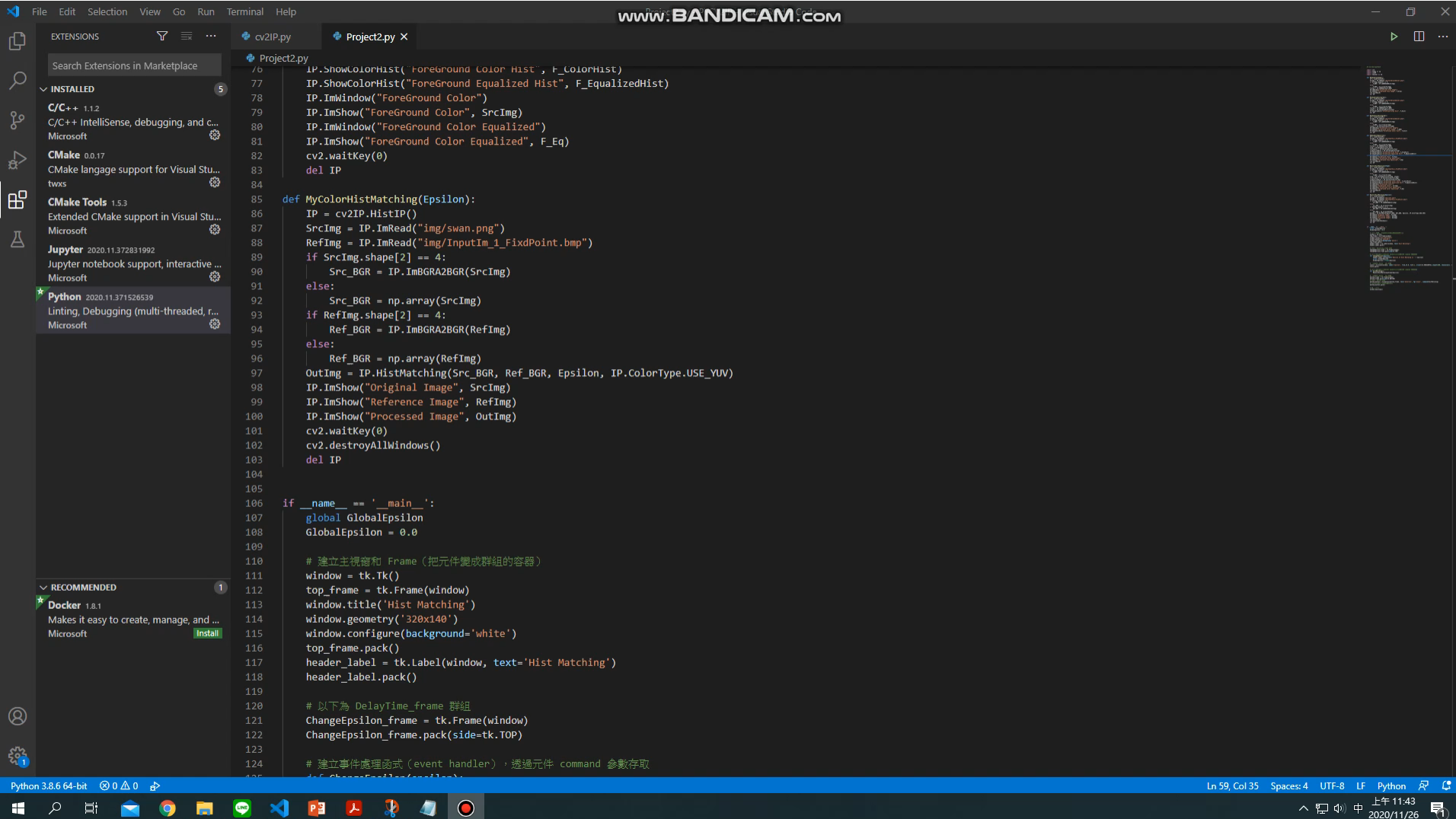
```
66 SrcImg = IP.imread( 'img/InpIm_1_FixdPoint.bmp' )
69 if SrcImg.shape[2] == 4:
70     F_BGR = IP.imbgra2bgr(SrcImg)
71 else:
72     F_BGR = np.array(SrcImg)
73 F_Eq = IP.ColorEqualize(SrcImg, CType)
74 F_ColorHist = IP.CalcColorHist(SrcImg)
75 F_EqualizedHist = IP.CalcColorHist(F_Eq)
76 IP.ShowColorHist("Foreground Color Hist", F_ColorHist)
77 IP.ShowColorHist("Foreground Equalized Hist", F_EqualizedHist)
78 IP.imshow("Foreground Color", SrcImg)
79 IP.imshow("Foreground Color Equalized", F_Eq)
80 IP.imshow("Foreground Color Equalized", F_Eq)
81 IP.imshow("Foreground Color Equalized", F_Eq)
82 cv2.waitKey(0)
83 del IP
84
85 def MyColorHistMatching(Epsilon):
86     IP = cv2.IP.HistIP()
87     SrcImg = IP.imread("img/swan.png")
88     RefImg = IP.imread("img/InpIm_1_FixdPoint.bmp")
89     if SrcImg.shape[2] == 4:
90         Src_BGR = IP.imbgra2bgr(SrcImg)
91     else:
92         Src_BGR = np.array(SrcImg)
93     if RefImg.shape[2] == 4:
94         Ref_BGR = IP.imbgra2bgr(RefImg)
95     else:
96         Ref_BGR = np.array(RefImg)
97     OutImg = IP.HistMatching(Src_BGR, Ref_BGR, Epsilon, IP.ColorType.USE_RGB)
98     IP.imshow("Original Image", SrcImg)
99     IP.imshow("Reference Image", RefImg)
100     IP.imshow("Processed Image", OutImg)
101     cv2.waitKey(0)
102     cv2.destroyAllWindows()
103     del IP
104
105 if __name__ == '__main__':
106     global GlobalEpsilon
107     GlobalEpsilon = 0.0
108
109     # 建立主視窗和 Frame (把元件變成群組的容器)
110     window = tk.Tk()
111     top_frame = tk.Frame(window)
112     window.title('Hist Matching')
113     window.geometry('320x140')
114     window.configure(background='white')
115     top_frame.pack()
116     header_label = tk.Label(window, text='Hist Matching')
```

The bottom status bar indicates the file is at line 76, column 59, using UTF-8 encoding, with the Python interpreter set to 'Python 3.8.6 64-bit'.

獨特設計之處(HSV 色彩空間)



獨特設計之處(YUV 色彩空間)



The screenshot displays a Windows desktop environment. The primary application is Visual Studio Code, which is open to a file named 'Project2.py'. The editor's interface includes a sidebar on the left with the 'EXTENSIONS' view active, showing a list of installed and recommended extensions such as C/C++, CMake, Jupyter, and Python. The main editor area contains Python code that performs color histogram matching. The code imports 'cv2' and 'IP' (likely ImagePy), reads two images ('img/swan.png' and 'img/InputIm_1_FixdPoint.bmp'), converts them to YUV color space, and compares their histograms using 'IP.HistMatching'. A GUI is created using 'tk.Tk()' and 'tk.Frame()' to display the results. The status bar at the bottom indicates the current cursor position is at line 59, column 35, and the file is encoded in UTF-8.

```
Project2.py
77 IP.ShowColorHist( Foreground Color Hist , F_ColorHist)
78 IP.ShowColorHist("Foreground Equalized Hist", F_EqualizedHist)
79 IP.ImWindow("Foreground Color")
80 IP.ImShow("Foreground Color", SrcImg)
81 IP.ImWindow("Foreground Color Equalized")
82 IP.ImShow("Foreground Color Equalized", F_Eq)
83 cv2.waitKey(0)
84 del IP
85
86 def MyColorHistMatching(Epsilon):
87     IP = cv2IP.HistIP()
88     SrcImg = IP.ImRead("img/swan.png")
89     RefImg = IP.ImRead("img/InputIm_1_FixdPoint.bmp")
90     if SrcImg.shape[2] == 4:
91         Src_BGR = IP.ImBGR2BGR(SrcImg)
92     else:
93         Src_BGR = np.array(SrcImg)
94     if RefImg.shape[2] == 4:
95         Ref_BGR = IP.ImBGR2BGR(RefImg)
96     else:
97         Ref_BGR = np.array(RefImg)
98     OutImg = IP.HistMatching(Src_BGR, Ref_BGR, Epsilon, IP.ColorType.USE_YUV)
99     IP.ImShow("Original Image", SrcImg)
100    IP.ImShow("Reference Image", RefImg)
101    IP.ImShow("Processed Image", OutImg)
102    cv2.waitKey(0)
103    cv2.destroyAllWindows()
104    del IP
105
106 if __name__ == '__main__':
107     global GlobalEpsilon
108     GlobalEpsilon = 0.0
109
110     # 建立主視窗和 Frame (把元件變成群組的容器)
111     window = tk.Tk()
112     top_frame = tk.Frame(window)
113     window.title('Hist Matching')
114     window.geometry('320x140')
115     window.configure(background='white')
116     top_frame.pack()
117     header_label = tk.Label(window, text='Hist Matching')
118     header_label.pack()
119
120     # 以下為 DelayTime_frame 群組
121     ChangeEpsilon_frame = tk.Frame(window)
122     ChangeEpsilon_frame.pack(side=tk.TOP)
123
124     # 建立事件處理函式 (event handler) , 透過元件 command 參數存取
125     def ChangeEpsilon(event):
```

組員分工表

| 組員 | 工作分配比重 | 內容 |
|-----|--------|-------|
| 李宗晏 | 100% | 程式、報告 |

Thanks for your attention

