Digital Image Processing

Term Project

2022 spring NCHU EE COURSE

4107062045 資工四 陳冠軒

Table of contents:

1. Introduction
2. Method description
3. Results comparison & discussion
4. Conclusion
5. References

**Introduction**

今天這個報告主要是根據Progressive color transfer for images of arbitrary dynamic range這篇paper所提出的演算法進行實作，這篇paper的目標是將輸入的source image透過輸入的target image所提取出的顏色特徵再調整source image histogram的方式，達到將source image的顏色相似於target image的結果。

論文所提及的方法是會先將輸入的圖片轉到CIELab空間，然後再執行演算法，對各自的channel修改histogram然後再合併，最後輸出結果，這篇論文的特色是他會根據不同尺度下的histogram進行修改，這樣的方法，可以讓原始圖片的調色更接近目標圖片的調色，而且這個方法不需要使用者輸入很多參數，讓使用者可以輕易達到調色的效果。

**Method description**

我的程式是利用python撰寫，都在colab上執行，會使用到opencv所提供的一些module，因此事先都要先import好，在完成環境設定後，我先定義一些函式(function)，如下:

GetTransformedImage(source,des\_images,filename)

changeColor(source,channel,dataPath,des\_images)

Hist\_multi\_scales(Hs,Ht,Vs,Vt,Bin,Cs)

Find\_peaks(H)

Hdrhist(I,BINS,minV,maxV)

Hist\_match(Iin,Ho,vs)

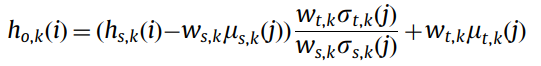
Histogram\_Transfer(Hs,Ht,weights)

Hdrhist()會根據給的Image和Bins數目建構出histogram

Find\_peaks()會用觀察二次和一次微分的結果找出histogram的轉折點

在GetTransformedImage()裡，先將原圖分成三個channel(L\*a\*b\* color space)再丟入changeColor()裡執行調色，最後才會將每個channel調色之後結果合併，再轉回RGB color space，輸出成圖片。

在changeColor()裡會先建構出source histogram和target histogram，之後再放入Hist\_multi\_scales()裡針對不同scales的histogram做histogram\_Transfer()。

histogram reshape是根據這個公式實現。

會針對每個scales做reshape，讓部分特徵的顏色相似，在做完每個scales後，會再根據原始的target最後再做一次reshape讓整體histogram可以更對齊。

***My implementation(github link)***

<https://github.com/jemmy9211/DIP_HW/blob/main/DIP_final.ipynb>

**Results comparison & discussion:**



Figure 2 : target image



Figure 4 :

Other result

Top is source

Mid is target

Bottom is result

Figure 1 : source image

Figure 3 : my result image



Anthor experiment result

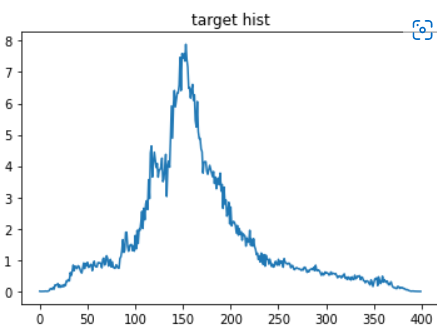
Other result

Top is source

Mid is target

Bottom is result

Figure 5: Source and target histogram



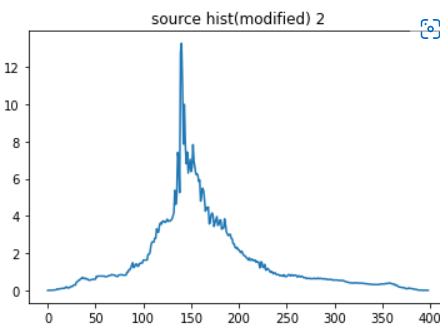
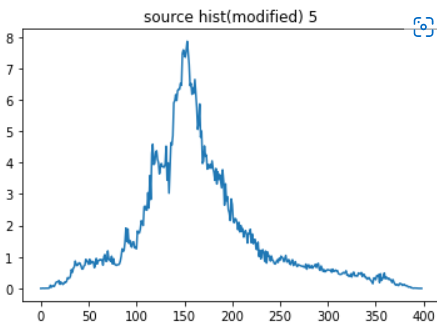
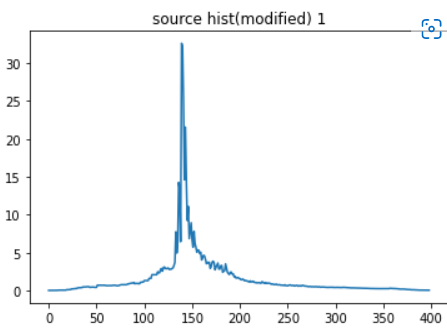
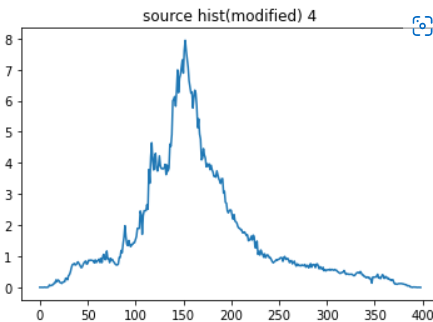
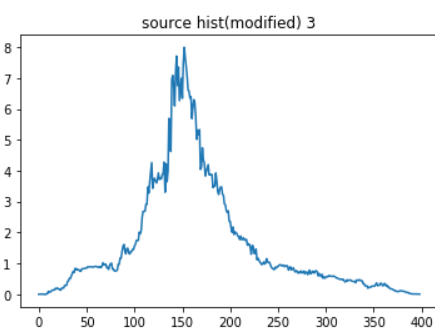
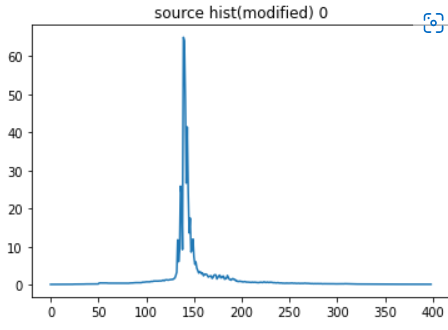
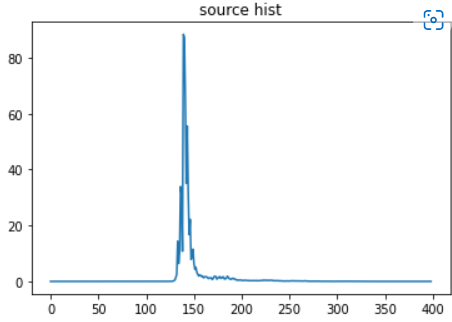


Figure 6: The transformed histogram at different scales (only one channel)

**實驗結果討論:**

Figure 1、2、3分別展示原始source和target以及我使用程式碼跑出來的結果，因為Figure 2有顏色很多樣，因此調色後的結果會讓Figure 1的顏色變得更鮮豔，而第二個的結果因為target的色調相對沒那麼多樣化，因此結果就不會是呈現鮮豔的結果，但是能有效的把target的色調轉移到source上面，讓source image從原本夏天的風格轉化成有點秋天傍晚的感覺，因為target image正是晚霞的感覺。

我也挑出在執行Figure 1時的其中一個channel的histogram，並且用matplotlib繪製成圖表，在Figure 5中左邊是Figure 1的其中一個channel的histogram，右邊是Figure 2的histogram，丟進程式執行後，根據論文所提及的演算法計算出所得到的histogram output(Ho)，我輸出每個尺度(scales)的Ho，當weight越大，histogram會越接近target image，當weight越小，histogram會越接近source histogram，結果如Figure 6所示，這樣分開調整各個尺度的hisgram的方法，可以使粗略和精細的圖片特徵分開考慮，因此所得的結果表現更好。

**Conclusion**

我認為這篇paper所提及的方法和以往的color transfer method都不同，他是利用histogram的方法讓原始的色調相似於目標色調，而且還考慮了不同scales的情況，讓圖片細部的部分顏色可以更精準，我在這次實作的過程中，也學習到很多在影像處理上的技巧，剛開始有很多程式不太知道怎麼寫，但透過大量上網爬文和參考別人github的code我也漸漸學習到一些影像處理的技巧，最後把這篇paper的功能實作出來，是非常有成就感的，看到一個理論真實的可以應用在現實世界中，讓我熱血沸騰。

**References**

[1] Tania Pouli, Erik Reinhard, Progressive color transfer for images of arbitrary dynamic range, Computers & Graphics; 2011

[2] Safithtechi, Emo\_Filters(<https://github.com/safithetechi/Emo_Filters.git>); 2020