基於 Opencv 的相片動畫水彩化演算法之優化

NTU Virtual Reality, 2018 Spring Final term project

Teammates

B04902028 洪浩翔
B04902090 施長元
B04902112 張凱捷

Introduction

電腦視覺領域中，將圖片風格化是很常見的應用。本組先前專注於 survey 將圖片水彩化的相關研究，找到幾個相關的方案；但這些方案大多成果不太理想，或是效率不佳，有時單張圖片的運算時間就需要數秒甚至更多的時間進行風格轉換。

因此，我們基於這些使用 OpenCV 的水彩畫算法，針對了多個方向進行優化，使得水彩化的效率可以達到近乎 Real time，甚至我們將此算法推廣至 GIF 動畫進行轉換，使用我們的算法可以讓 20 個畫格的動畫在 1.5s 以內完成轉換風格，加上合併 GIF 的時間也可以在 5s 以內轉換完成

(Based on Ryzen 5 1600 with stock frequency)。

Implementation

- GIF to JPG
  使用 Linux 的指令 convert 將 gif 切成一個一個的 frame。

- 水彩化
  首先，先用 Gaussian blur 將原本圖畫 I1 處理一次，然後再透過簡單的除以 k 再乘上 k 的方法做 quantization 來減少色彩，然後再做一次 Gaussian blur，得到一張模糊且減色過的 image I2 來模擬水彩的感覺。

  為了模擬水彩畫中物件的邊緣，我們接著將 I1 做 Gaussian blur 然後轉灰階，然後用 sobel edge detection 跟 canny edge detection 兩種方法來實作抽取邊緣的方法並比較，以此方法得到的圖片為 I3。接著將 I3 合成到 I2 上來增加邊緣的實感，然後再調整此結果與 I2 的比例，合成出最終的圖像 I4。

- JPG to GIF
  使用 imageio 將轉換過的 frame 合成一個新 gif。

Difficulty & Solution

- k-means
  在原作中，作者使用 k-means 來實作減色的挑選，然而 k-means 是個非常耗工費時的方法，跑一張圖在這部分會消耗絕大多數的時間，若圖比較大的話，有可能完成一張圖要幾分鐘才行，因此我們決定將這部分視為達成 Real time 的 bottleneck，並針對此部分進行優化。

  在比較結果過後，發現 k-means 並不一定會比直接做除法再乘回去的方法來得實用，單純用乘除運算的結果在大多數狀況下都可以很接近使用 k-means 分類的結果，因此取而代之，我們使用矩陣的除法以及乘法來實作 quantization 減色，如此即可更靠近 Real time。

  而矩陣的乘除法一直以來是電腦領域很重視的加速領域，因此乘除法我們使用 python 中的 numpy 進行加速。

- Edge detection
  為了要做到水彩畫中物件邊緣的線條感，所以我們必須選擇要使用哪種 edge detection 的方法，起初我們使用了 sobel edge detection，但是結果並不好，因為 sobel 太容易被 noise 所影響。

  為了避免 noise 的影響，所以不論單純 first order 或 second order edge detection 都未必能滿足我們的需求，所以最後我們使用 canny edge detection 來實作抽取邊緣的部分；canny 能容忍較多的 noise，表現也較其他的 edge detection 方法好，就結果而言，canny 也能正確的取出 edge，所以 edge detection 的問題也在使用 canny method 後完美解決了。

- GIF and JPG
  原先我們使用 python Pillow 中的 PIL.image 來切割 GIF，但是編碼與色彩的不同造成有些 GIF 轉出來的圖片有一些問題 (這是一個 open issue)，因此我們最後放棄使用這個套件，使用 subprocess 直接利用命令列 call convert 這個 unix 的指令進行 GIF 切割成 JPG。

Source Code

https://github.com/shihehe73/WatercolorGIF

Result

文字敘述的關係，並沒辦法放入 GIF 呈現我們的結果，煩請至我們的 Github 上(Source code section above) 觀看我們所轉換的 GIF 範例。

原圖：

轉換後：

原圖：

轉換後：

原圖：

轉換後：

原圖：

轉換後：

Reference

- Python: 写真を水彩画風に変換する(We refine the algorithm based on this code.) http://hibari1121.blog.fc2.com/blog-entry-105.html
- Change digital image into a watercolor style https://github.com/auroraka/waterColor
- Parallel Style Transfer http://jaisrael.github.io/ParallelStyleTransfer/