Acceleration of Poisson Image Cloning

Approach 1: Successive over-relaxation method

我首先嘗試了 PDF 內的作法，將前幾個 iteration 的 w 值設為 2，但得到的結果卻不怎麼理想，在邊界出會出現許多交錯的黑白格子，如下圖。而格子靠近中央白色部分的點會較偏向天空的藍色，但若是繼續 iteration 下去則會整張圖跑掉。因此這個做法對於整個演算法的加速並沒有很大的幫助。

觀察發現過大的 w 值會導致以上所述的問題，因此改用較小的 w 值。實際操作之後發現較小的 w 值會大幅加速顏色推進速度，但卻會在色塊中產生許多空格，而在演算法進行中，由於本圖中有大量的白色，根據 update 的式子，下一刻的顏色將由周邊點的顏色平均，因此藍白格子會在 iteration 之間交錯，而藍色格子會隨著 iteration 次數變深。如下圖

我也嘗試過在最後一次 iteration 將 w 改成 0.5，平均藍白兩色格子的內容，但這樣執行結果跟不使用 SOR 相差不多。

Approach 2: Start from background

由於上個方法主要遇到的問題在大片空白的部分，而會出現大片空白是因為我們執行 Image cloning 的開頭是由 target image 開始的，同時觀察式子可以發現 new term 主要是由 background 組成的，因此試著由 background 開始。實驗結果如下圖，在 10000 個 iteration 的實驗上有顯著的改善。

使用 Background 圖 (background10000.png):

使用 Target 圖 (Target10000.png):

Approach3 scaling:

此處使用的 scaling 是 scale 成 1/4(4 個 pixel 合成一個)。單純 scale 縮小又放大並貼到原圖的圖片如下圖，可以看到解析度明顯變差。

Scale.png:

接下來我們將 scale 套用在 poisson image cloning。先將 target, mask, fixed 都 scale 變小之後執行 PoissonImageCloningIteration 重複 5000 個 iteration。由於收斂的時間正比於圖片大小，因此 5000 個 iteration 將是 scale 後圖片的理論收斂時間。之後我們將處理完的目標區快重新放大，並當作初始值丟到正常大小的 PoissonImageCloningIteration 並重複 5000 個 iteration。得到結果如下圖:

Scale25002500:

相較第二頁 10000 iteration 的圖有明顯改善，甚至勝過 15000 iteration 的結果。