GPGPU Assignment #3

Acceleration Part

b02901065 電機四 李洛曦

在加速部份，我採用的是hirarchical method，先將mask與target縮放(sampling)至1/8倍成mask_1, target_1，利用縮放後的mask_1與target_1對背景做固定值的運算，接著使用Jacobi Iteration運算數次後，將target_1的結果以及mask_1使用nearest-neighbor插值法放大兩倍，為result_2與mask_2，此時將target再次縮放為1/4倍成target_2，使用target_2與放大後的mask_2去做固定值的運算，並將result_2代入Jacobi Iteration Solver，持續執行到原尺寸時，改為使用target與放大後的mask_4做固定值的運算，使從較小的圖片放大後的結果能夠依照原圖gradient的分佈將nearest-neighbor method帶來的效應消除，執行完最後一次運算後，將得到的結果與背景透過縮放後的mask合併在一起。

在實作的過程中，原本在每次縮放時都會將mask重新縮放為該大小，但是由於插值放大、經過著色修改後的result與從原圖縮小的mask可能會產生區域不一致的問題，因此後來改為mask由前次執行的尺寸依照同樣插值法放大而來。

在該執行平台上，使用Naive Jacobi Solver約執行20000次達到收斂，執行時間為7.86秒：

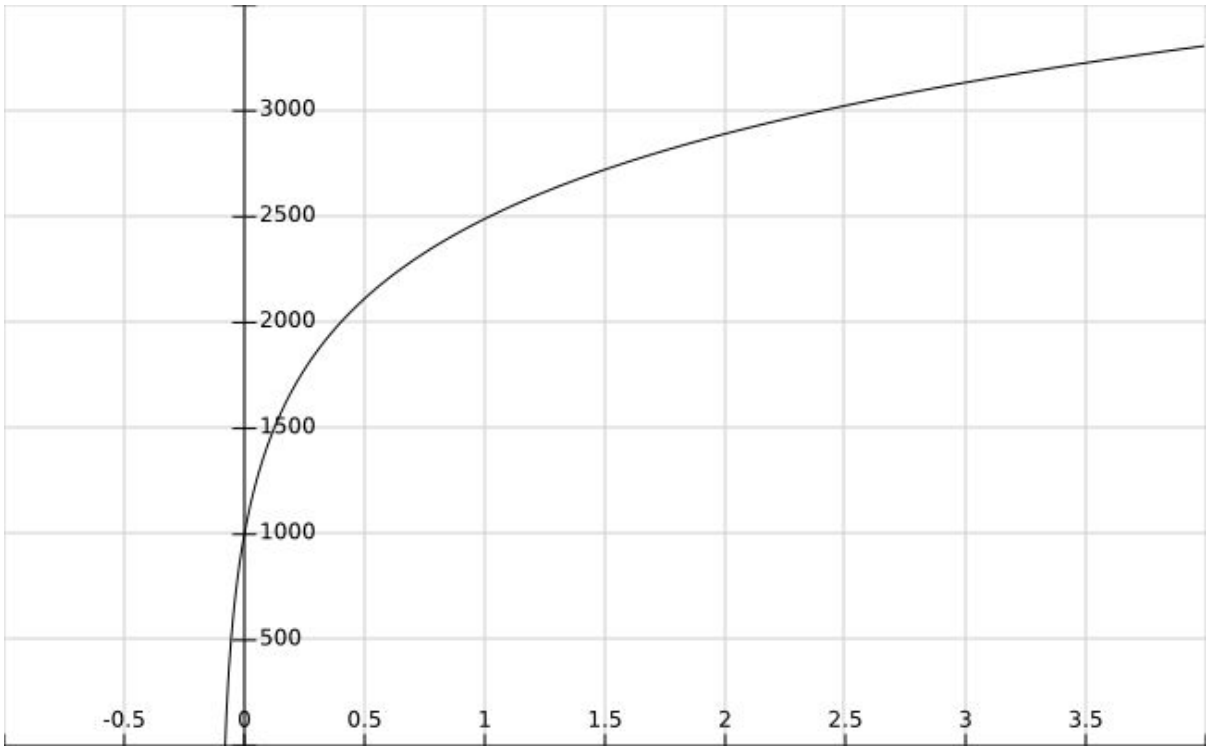
在Iteration次數上，我分別使用了線性遞增，線性遞減以及log函式來決定每一個scale階段的執行次數：

由上面兩張圖可以看出來，使用遞增函數的效果較好，接著使用斜率為1000執行，結果如下：

在使用log函數時，為了決定基底，先使用1/8單輪來試驗收斂所需的次數，經實驗後，可以得知在1/8倍時約1000次達到收斂：

在經過幾次的試驗後，採用下列log函數：

$$1000 \times (log_5(iteration\_index \times 10 + 1) + 1)$$



即在i==0時，次數為500，i==3時，次數為1566次，得到的結果如下：

上述方法的總執行次數以及所需時間：

方法	總次數	所需時間(秒)	圖片大小加權次數
Naive	20000	7.86	20000
Linear increasing	10000	1.98	4890.625
Log increasing	9510	1.57	4025.625

比較三者的結果：

可以發現在Naive的情形下，中間區域還是有點白色，並不算完全的收斂，而後兩者的表現就相去無幾。