

動機

我們是對loli特別有愛的團隊，我們想要讓loli從圖片中分割出來。為了達到這個目標，我們必須了解並且剖析Image matting的各種演算法，並且比較不同做法的異同，找到一組最適合把loli分割出來的方法，並且把loli放到最適合他的環境。我們希望讓使用者可以指定想要替換的背景範圍，並且把这个背景範圍去除，然後加入使用者喜歡的新背景，並與loli產生新的圖片。

方法

A Bayesian Approach to Digital Matting

Yung-Yu Chuang, Brian Curless, David Salesin, and Richard Szeliski
CVPR 2001

$$C = \alpha F + (1 - \alpha)B$$

C:composite

α :alpha channel

F:foreground

B:background

C是原圖， α , F, B是未知數，要算出每個pixel中，前景和背景的比例是多少，如果全部都是前景， $\alpha=1$ ，如果全部是背景 $\alpha=0$ ，如果前景部分有點透明，那 α 介於0~1之間。

$$\log P(F) = -(F_i - \bar{F})^T \Sigma_F^{-1} (F_i - \bar{F}) / 2$$

contribution of a nearby pixel p_i :

$$w_i = \alpha_i^2 g_i$$

$$\bar{F} = \frac{1}{W} \sum_{i \in \mathcal{N}} w_i F_i$$

$$\Sigma_F = \frac{1}{W} \sum_{i \in \mathcal{N}} w_i (F_i - \bar{F})(F_i - \bar{F})^T$$

$$W = \sum_{i \in \mathcal{N}} w_i$$

在要解的pixel周圍框一塊區域，統計一下F的分布長什麼樣子(R, G, B)。

pixel i的 權重 w_i 參考 α_i^2 和距離 g_i ，W是權重總和，

算出F的加權平均，然後normalize。

算出F的covariance matrix，也是用加權的方式算。

對background也做。

然後開始iteration解 α , F, B。

一開始的 α 設成trimap中pixel(a, b)鄰居（九宮格中間那格以外的其他八格）的平均，然後重複做以下步驟：

1. 固定 α ，去解 F, B ：

$$\begin{bmatrix} \Sigma_F^{-1} + I\alpha^2/\sigma_C^2 & I\alpha(1-\alpha)/\sigma_C^2 \\ I\alpha(1-\alpha)/\sigma_C^2 & \Sigma_B^{-1} + I(1-\alpha)^2/\sigma_C^2 \end{bmatrix} \begin{bmatrix} F \\ B \end{bmatrix} = \begin{bmatrix} \Sigma_F^{-1}\bar{F} + C\alpha/\sigma_C^2 \\ \Sigma_B^{-1}\bar{B} + C(1-\alpha)/\sigma_C^2 \end{bmatrix}$$

[6-by-6] [6-by-1] = [6-by-1]
直接用左除處理。

2. 算出新的 α ：

$$\alpha = \frac{(C - B) \cdot (F - B)}{\|F - B\|^2}$$

然後跟原本的 α 比較，如果差異很小就不繼續做了，否則繼續iteration，直到超過預先設定的最大iteration數。

iteration結束後得到 α, F, B 。

由於在implement的時候，我們不知道 σ_c 究竟要放什麼值比較好，又找不到比較general的解法，所以只提供三張比較好的結果在Main.m，每次run section的output是Bear.png。

實驗結果

由於我們不知道 σ_c 到底要放什麼值，所以基本上採取顏色越多， σ_c 越大的方式。
output在Bear.png。

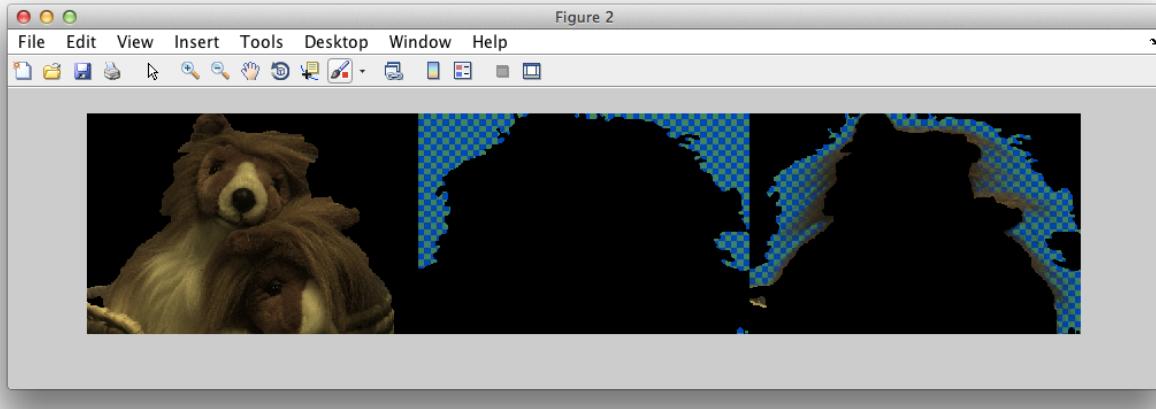
Main.m產生的結果

$\sigma_c=8$

iteration=10

原圖	新背景（使用者可以選）	output
		

前景、背景、不知道是前景還是背景的部份



$\sigma_c=10$

iteration=20

原圖	新背景（使用者可以選）	output

前景、背景、不知道是前景還是背景的部份

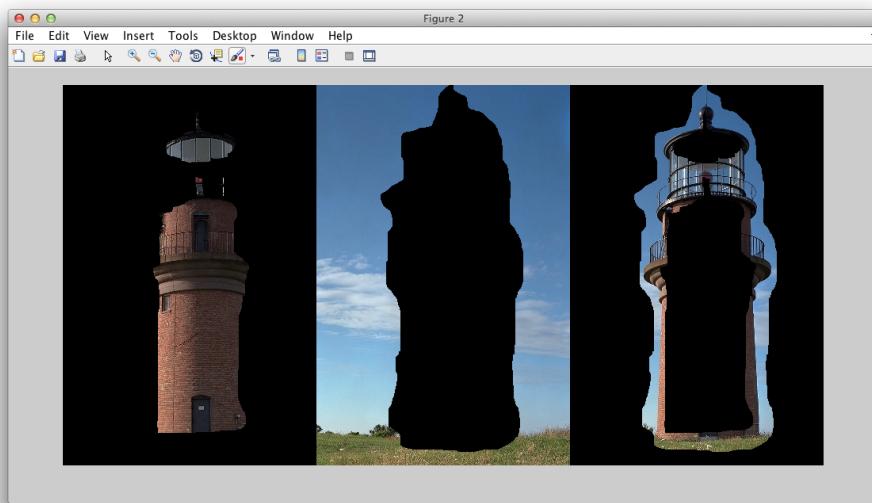


image matting

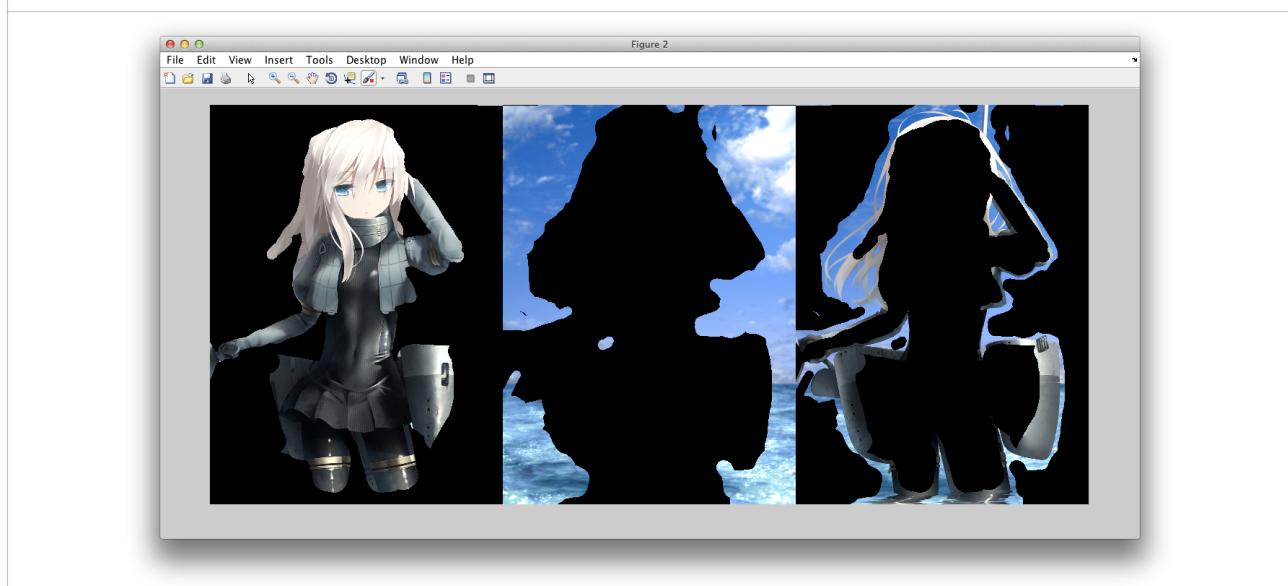
loliLoliHunter(>_<)

$\sigma_c=10$

iteration=20



前景、背景、不知道是前景還是背景的部份



比較

調整 σ_c 與iteration比較是否會影響結果。
為了方便觀察，背景都選擇彩虹。

第一張圖





這張圖的背景非常簡單，只有兩種顏色。

$\sigma_c = 8$, iteration = 10



image matting

loliLoliHunter(>_<)

$\sigma_c = 8$, iteration = 100



$\sigma_c = 8$, iteration = 1000



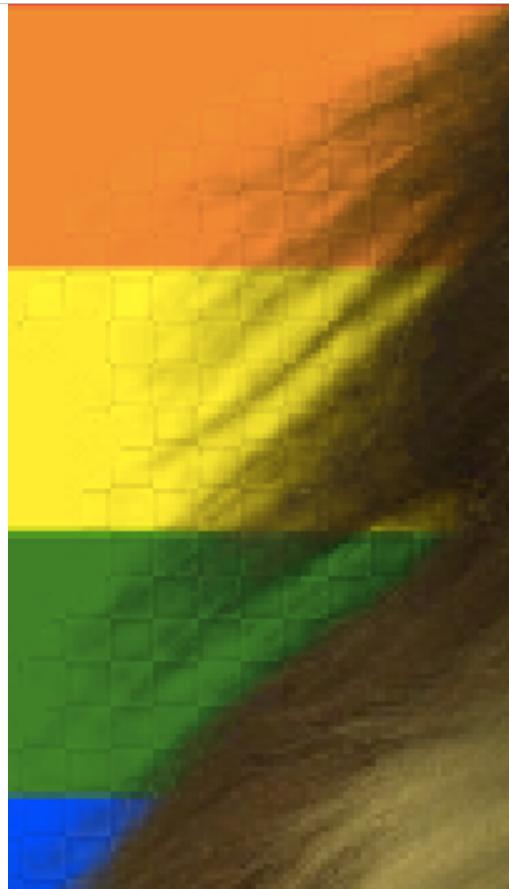
因為背景非常單純，所以即使加大iteration數，改變也不明顯， $\sigma_c = 8$ iteration = 10 和 $\sigma_c = 8$ iteration = 1000 的圖相減會得到一張幾乎是黑色的圖。

$\sigma_c = 8$, iteration = 10



雖然背景非常單純，但可能是implement問題或是這個算法本身不夠完善，可以發現在玩偶的毛四周仍有許多格子背景沒有被去除。

$\sigma_c = 8$, iteration = 10



由於我們不知道正確的 σ_c 值該怎麼取，所以我們也自己去猜 σ_c 值。

但是由於這張圖的背景非常單純，所以結果差不多，將兩圖相減取絕對值也是一片黑，幾乎一樣。

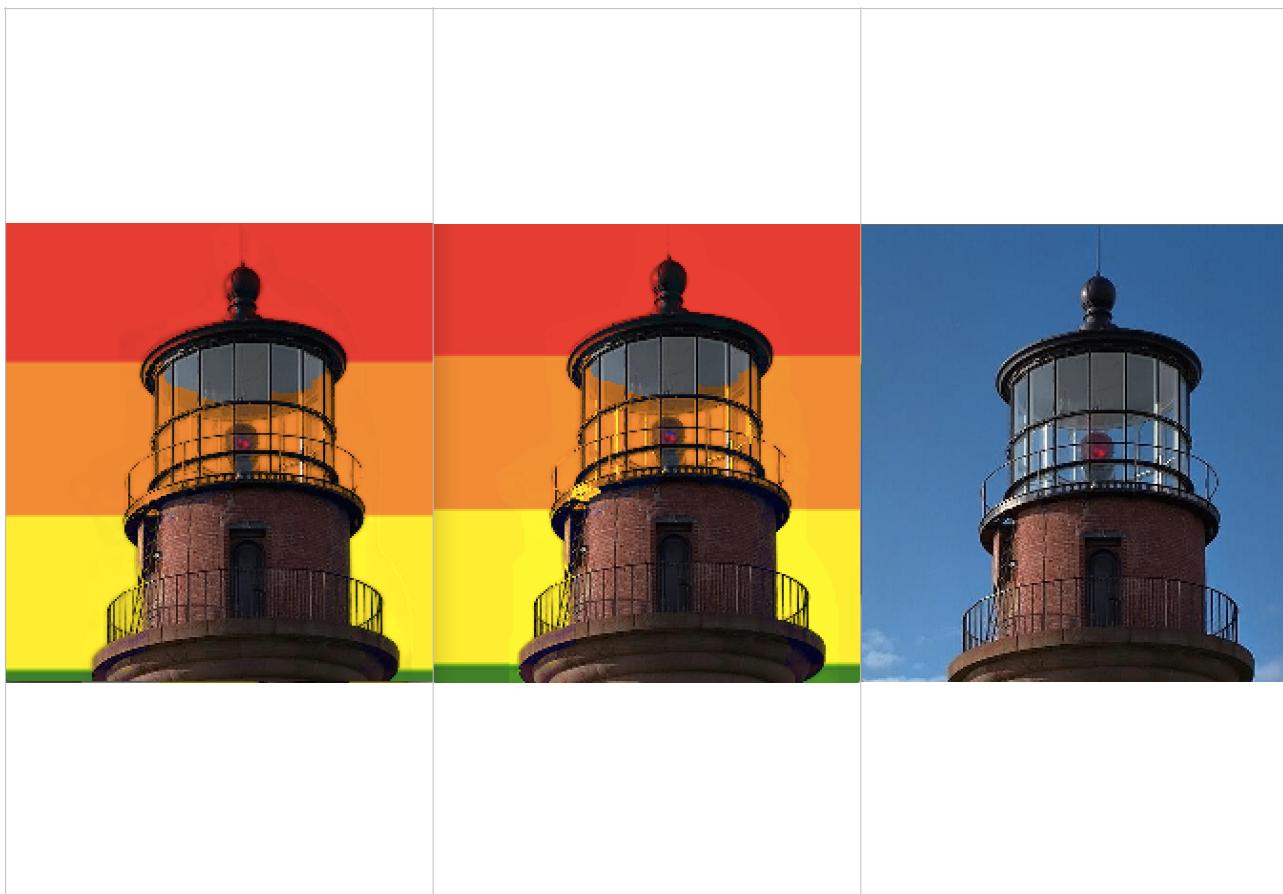
$\sigma_c = 4$, iteration = 10



第二張圖



$\sigma_c = 10$	$\sigma_c = 10$	原圖（局部）
-----------------	-----------------	--------



可以發現原圖中反光的部份隨著iteration增加而判斷出來。



可能是因為在原圖中的面積比較大，導致iteration變大時，被判斷成前景的部份較多。



可能是因為原圖背景中主要色調有藍色、白色、綠色，並不如第一張圖（玩偶）那麼簡單，所以調低 σ_c 可能會造成被判斷成前景的部份增加。



image matting

loliLoliHunter(>_<)



但是維持 $\sigma_c = 5$ ，加大iteration又會發現結果比iteration數小的時候好，所以iteration增加可能具有修正的作用。

放大塔底，可以發現跟毛一樣，草的邊緣有部分被留在前景的塔底了：

$\sigma_c = 10$ ，



第三張圖

我們找到一張loli的CG。
由於CG的前景和背景差異不大，

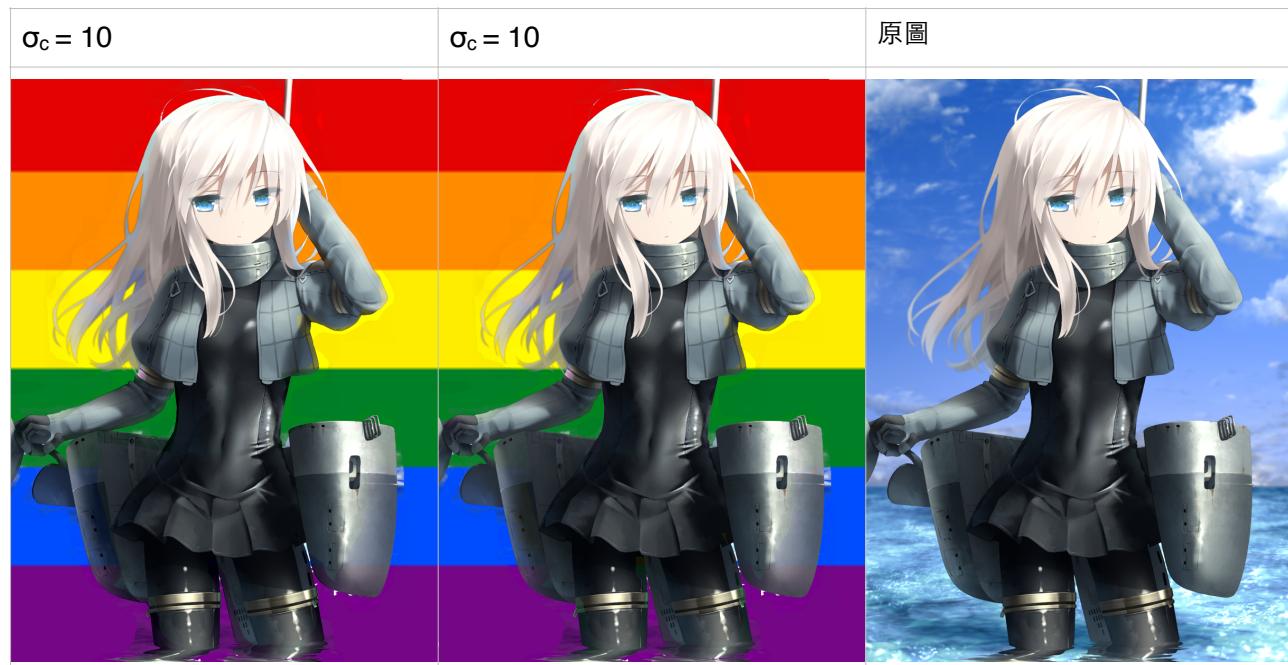
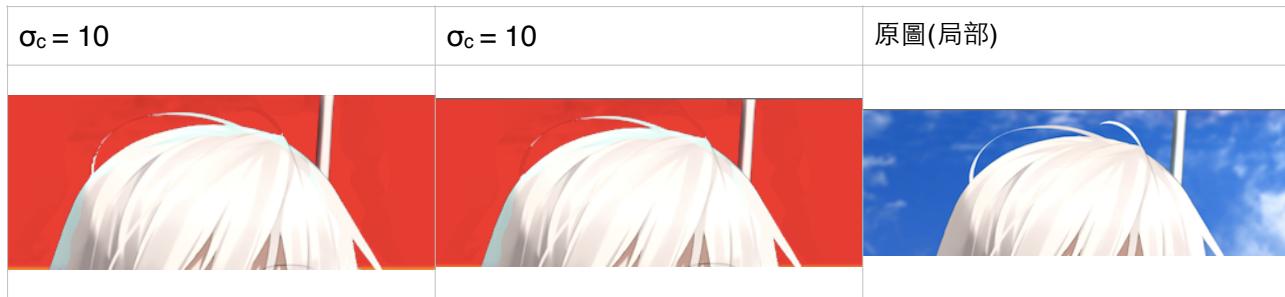
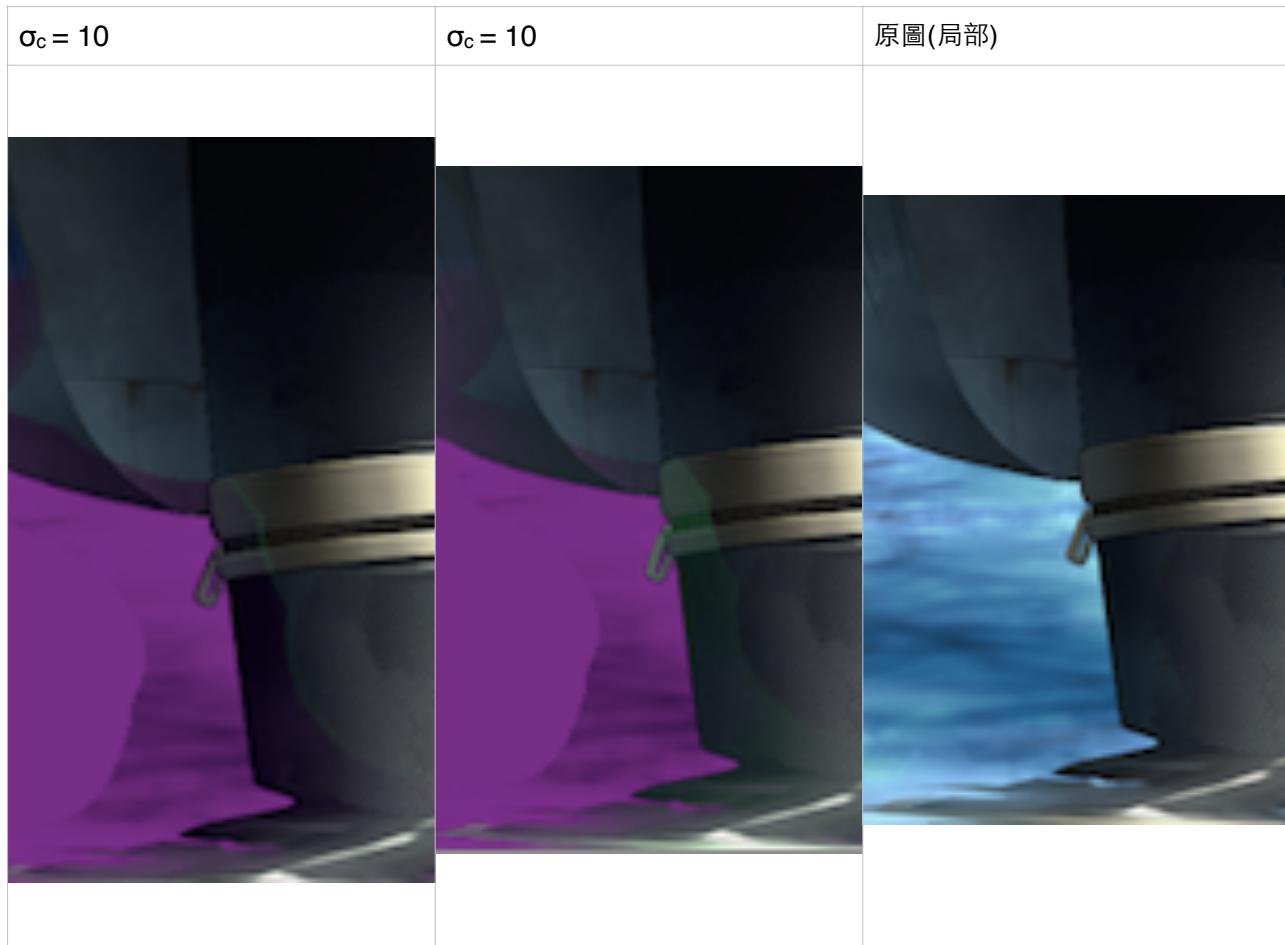


image matting

loliLoliHunter(>_<)



隨著iteration的增加，面積小又和背景有點相似的呆毛有部分消失。



大腿部分有些藍色閃光，在iteration增加後變得明顯。

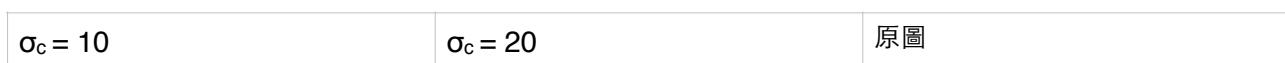
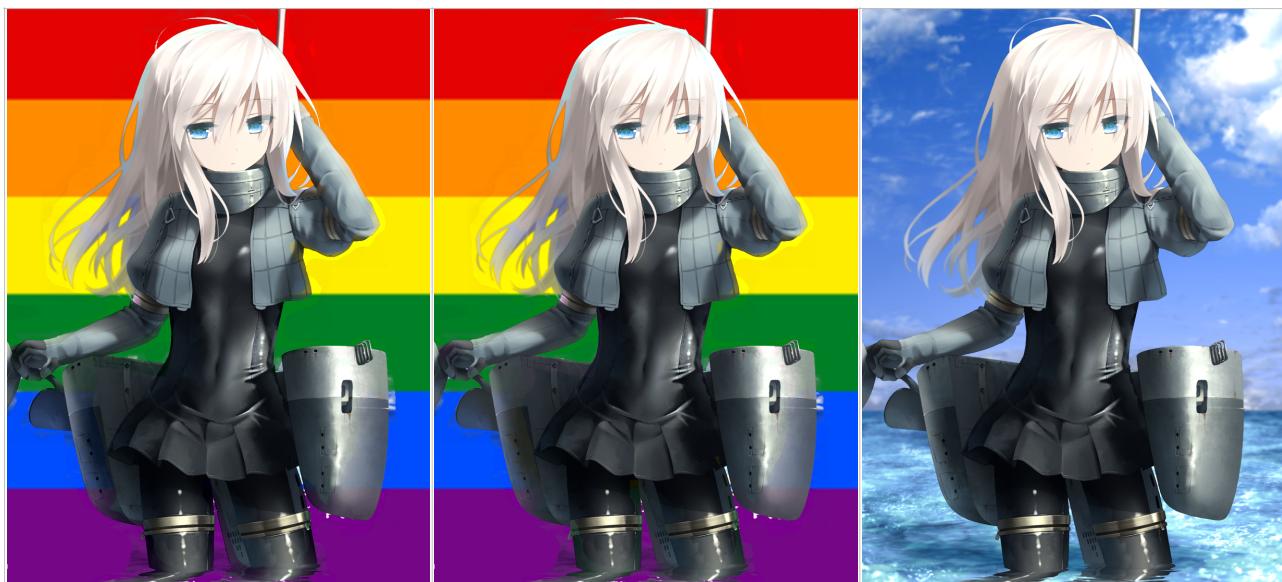


image matting

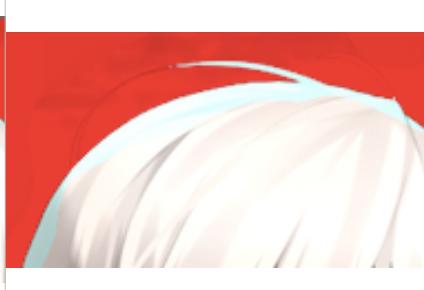
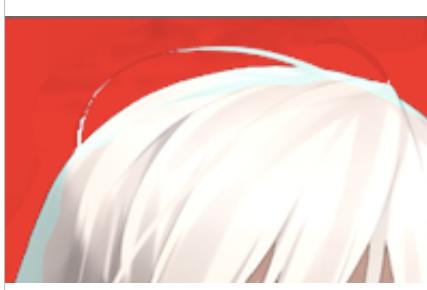
loliLoliHunter(>_<)



$\sigma_c = 10$

$\sigma_c = 20$

原圖(局部)



$\sigma_c = 20$ 時，呆毛可能被當成是背景，因而有部分消失。

$\sigma_c = 10$

$\sigma_c = 20$

原圖(局部)



$\sigma_c = 20$ 時，藍色反光變得明顯。

結論

由於我們不知道 σ_c 的值該用什麼，只能憑感覺去決定 σ_c 的值，只從 σ_c 的意義上覺得背景顏色越多的話，應該要用比較大的 σ_c 值，我們可能沒有完全理解式子中的 σ_c 該放什麼，所以沒找到一個 general 的 σ_c 值算法（比方說什麼 matrix 的 determinant）。

iteration 增加的話，更能夠呈現出原本的前景，但整體看起來可能未必比較自然。

結論就是，這類算法可能還有很多需要優化的部份，如果要用可能還是 call API 比較方便。

組員的分工

林致民 presentation

邱政凱 coding

賴怡文 report

遇到的困難

由於機率學的不夠深，有些算式可能沒有完全理解，尤其是 σ_c 該用什麼值的部份，一直未能有定論，而這個部分又很難找到簡單易懂的說明。

學到的東西

覺得機率非常重要，對於機率又有更深刻的了解了。

創新的想法以及未來可能的發展

這個算法還有很多種優化，像是flash matting、closed form matting等等我們沒有實作的部分。

我們一開始打算實作一些GUI的工具方便使用者畫出背景部分，但是我們光是為了搞懂算法就花了很多時間，因此trimap要另外準備，如果實作GUI讓使用者畫背景部分，甚至使用橡皮擦，擦掉畫錯的部份，就會方便很多。

Reference

- [1] Yung-Yu Chuang, Brian Curless, David Salesin, and Richard Szeliski. A Bayesian Approach to Digital Matting. In CVPR, 2001. <http://grail.cs.washington.edu/pub/papers/Chuang-2001-ABA.pdf>
- [2] <http://ocw.nthu.edu.tw/ocw/index.php?page=chapter&cid=125&chid=1514>
- [3] http://lms.nthu.edu.tw/sys/read_attach.php?id=637510
- [4] http://lms.nthu.edu.tw/sys/read_attach.php?id=639209