Experiment Report

Name	吴彬睿	ID	3180200084
Title	Grab Cut	Date	2019/6/11

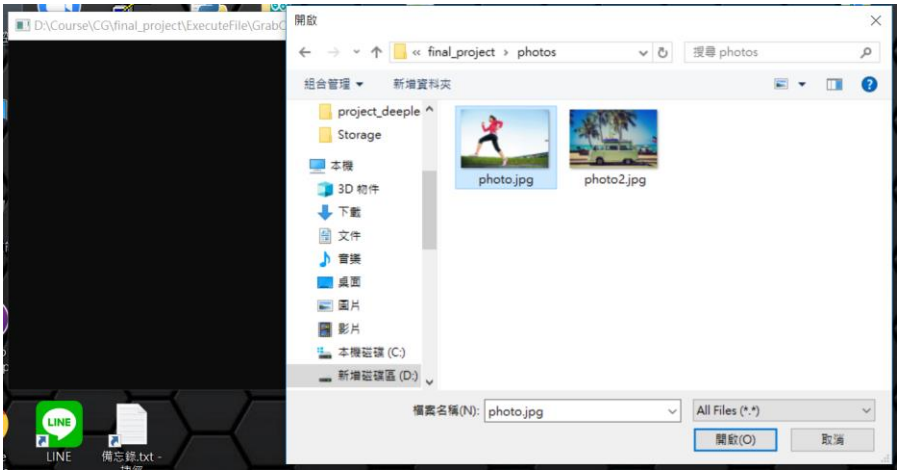
一、 实验内容、运行说明

1. 实验内容:

GrabCut 是一个经典的图像分割方法。它结合手工交互和自动的 Graph-Cut 算法来对静态图像进行高效的前景/背景分割。在这个项目中，你需要实现 GrabCut 算法的核心步骤，并完成简单易用的用户界面，用户界面要支持完成 GrabCut 所需要的基本交互。

2. 运行说明:

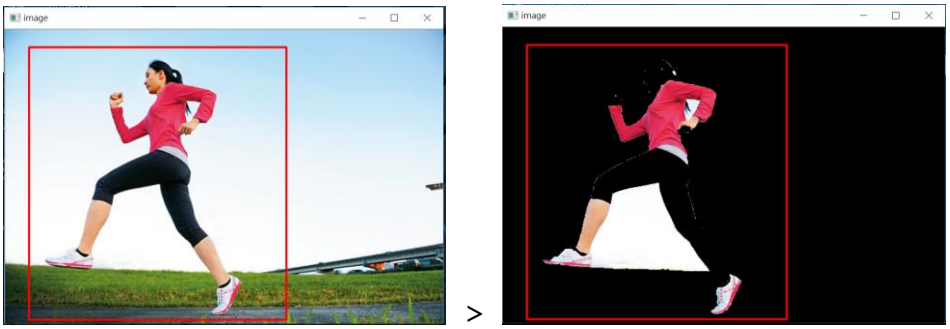
(1) 开启挡案



(执行挡案) 先选择要做 grab cut 的图片

Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

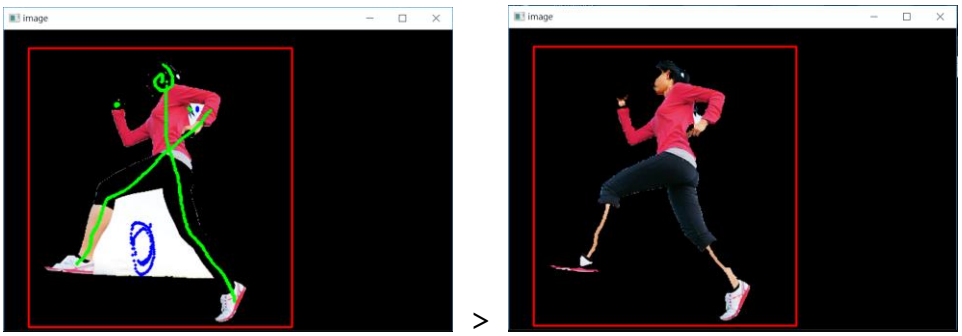
(2) 手动标注前景的筐筐



标注完成前景的位置后，执行一次 Grab Cut 迭代。

得到第一次 grab cut 后的前景。

(3) 辅助标示前景与背景、并进行第二次迭代:



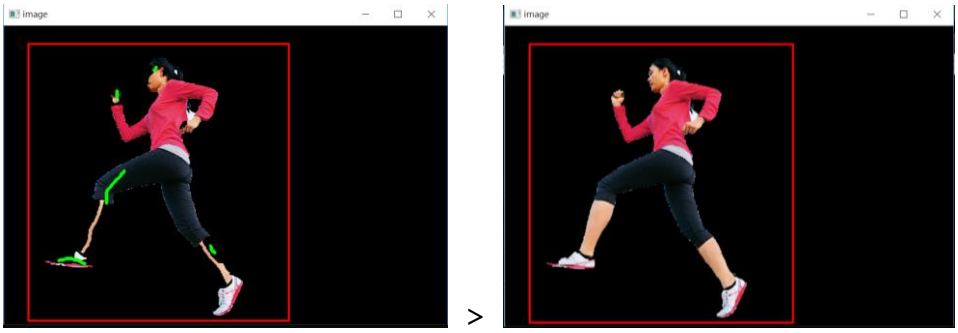
标示出一开始在 GMM 模型中被误认的，辅助电脑知道哪些属于前景，哪些属于真正的背景。

并在执行一次 Grab cut 的动作，得到一个新的分割。

可以看到在辅助的情况下，裤子成功被分类为前景，而皮肤色的地方还有点模棱两可，需要在迭代一次。

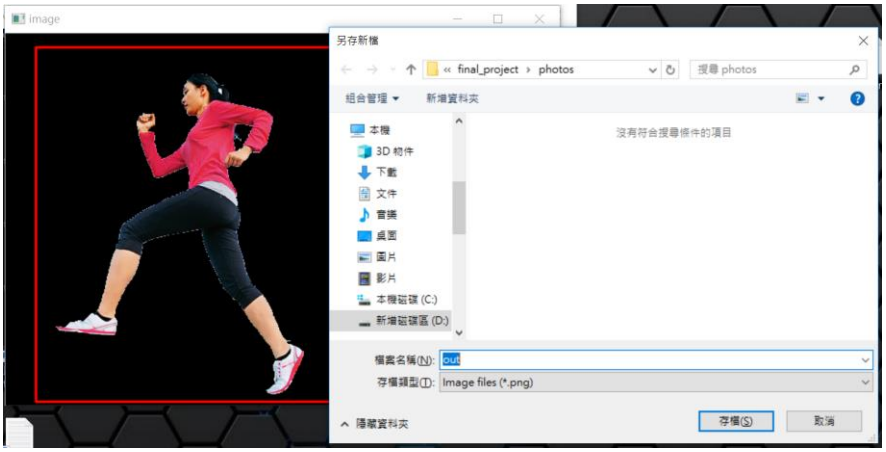
(4) 在进行一次辅助标示前景与背景、并进行第三次迭代

Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.



标示出鞋子、手部、脸部、腿等等皮肤色的地方，再进行一次 grab cut，可以看到结果已经非常完善。

(5) 输出结果



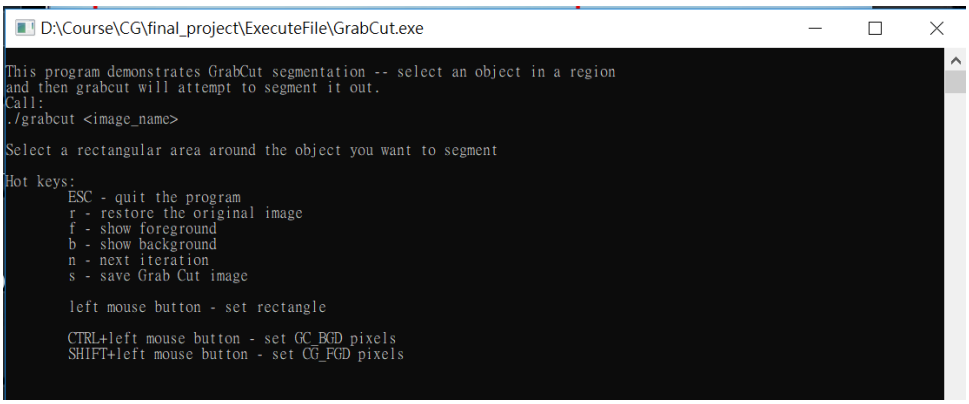
输出分割完成的结果，存成一个去背的挡案。

此档案一定要是 png 格式，因为只有 png 格式支援第四个通道 alpha 通道，也就是透明度通道，才能真正保存去背的效果，而 jpg 格式没有这个通道，所以存起来的图片背景会是白色的。

(6) 绘制想要的场景

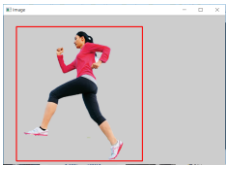
Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

3. 支援功能

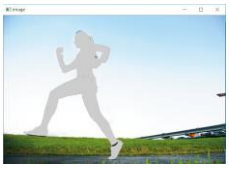


(1) r：还原图片

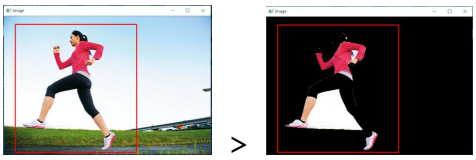
(2) f：显示前景



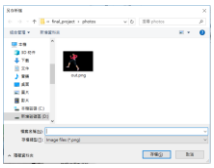
(3) b: 显示背景



(4) n: 进行一次 Grab Cut 迭代



(5) s: 存储去背的前景图片



Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

二、基本原理、细节

1. 基本原理:

(1) Kmeans

Kmeans 试一个在计算欧式距离下做 EM 的算法。

E-Step:

$$Q_i(z^{(i)}) := p(z^{(i)}|x^{(i)};\theta).$$

M-Step:

$$\theta := \arg\max_{\theta} \sum_i \sum_{z^{(i)}} Q_i(z^{(i)}) \log \frac{p(x^{(i)}, z^{(i)};\theta)}{Q_i(z^{(i)})}.$$

(2) GMM

GMM 是由 K 个高斯分布组成的:

$$p(x) = \sum_{k=1}^{K} p(k)p(x|k) = \sum_{k=1}^{K} \pi_k N(x|\mu_k, \Sigma_k)$$

E-Step:

计算出每一个 Sample 属于哪个 component。

$$\gamma(i,k) = \frac{\pi_k N(x_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_i|\mu_j, \Sigma_j)}$$

M-Step:

把每个 component 所属的 sample 做最大化似然估计，maximum-likelihood，得到每个 component 新的高斯模型。

$$\Sigma = \frac{1}{N_k} \sum_{i=1}^{N} \gamma(i,k)(x_i - \mu_k)(x_i - \mu_k)^T$$

(3) Minimum-Cut

Step1:

找出 S-t 的最大流，可以用 Fold Fulkerson 算法来计算。

Step2:

DFS 方法找出流量瓶颈。

Step3:

我們只找源點側點數最少的、匯點側點數最少的，其他的最小源匯割則不好找

Psuedo code:

```
void DFS(int i)
{
    visit[i] = true;
    for (int j=0; j<9; ++j)
        if (!visit[j] && F[i][j] < C[i][j])
            DFS(j);
}
void minimum_s_t_cut(int s, int t)
{
    // 求一個最大源匯流，源點為s點，匯點為t點。
    Edmonds_Karp(s, t);

    // 從源點開始遍歷，找出流量瓶頸。
    memset(visit, false, sizeof(visit));
    DFS(s);

    // 找出其中一個最小源匯割，會是源點側點數最少的最小源匯割。
    for (int i=0; i<9; ++i)          // 窮舉源點側的點
        if (visit[i])
            for (int j=0; j<9; ++j) // 窮舉匯點側的點
                if (!visit[j])
                    if (C[i][j] > 0)     // 要確定有邊
                        cout << "割上的邊有"
                             << "由" << i << "到" << j;
}
```

(4) Grab Cut:

a. 特点:

基于 Graph Cut 的实现，用高斯混合模型 GMM 代替灰度直方图，颜色模型取代单色图像，在 graph cuts 中只进行一次的最小化分割估计被 一个在估计和参数学习中可选的、可靠迭代的过程代替。对于 Trimap，用户只需提供不完全标记。

b. 吉布斯能量分布:

$$E(\underline{\alpha},k,\underline{\theta},z)=U(\underline{\alpha},k,\underline{\theta},z)+V(\underline{\alpha},z)$$

U:数据项

$$U(\underline{\alpha},k,\underline{\theta},z)=\sum_{n}D(\alpha_n,k_n,\underline{\theta},z_n)$$

$$D(\alpha_n,k_n,\underline{\theta},z_n)=-\log p(z_n\mid\alpha_n,k_n,\underline{\theta})-\log\pi(\alpha_n,k_n)$$

V:平滑向

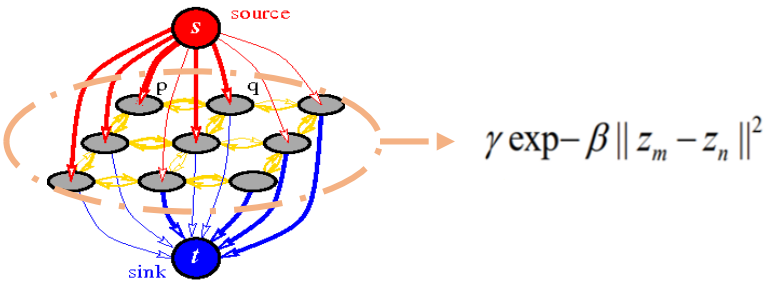
$$V(\underline{\alpha},z)=\gamma\sum_{(m,n)\in C}[\alpha_n\neq\alpha_m]\exp-\beta\|z_m-z_n\|^2$$

c. 求解最小化吉布斯能量分布:

刚好可以看成是一个作 Max-flow 解 min-cut 的问题。

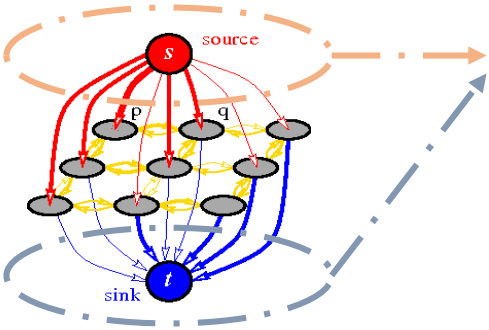
2. Graph 定义:

(1) 像素与像素之间的 edge w:



$$\gamma\exp-\beta\|z_m-z_n\|^2$$

(2) Source/Destination 与像素之间的 edge w:



$$-\log p(z_n\mid\alpha_n,k_n,\underline{\theta})$$

Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

三、遇到问题

定义 Graph 的部份，之前不知道 edge 的 weight 要怎么定义，最一开始就用一个固定值当做是 weight 但是做出来的效果不太好，想一想这样不太对，每个像素之间的 weight 不应该一样，这样不能分辨出前景与背景的区别，反而像素相似的应该有较大的 weight,而像素比较不相似的之间的 weigh 尽量要小，这样最后在做 min-cut 的时后比较容易选到这组边。后来仔细搜索了一遍就又平滑向的值当做试 edge 的 weight，做出来的效果是我想要的。

四、结果与分析

1. 结果:


>>


最终结果，我觉得还不错像这种构图简单的图片都分割的还不错，GMM 的 K 调 5 刚好可以体现出这个人，一类是黑色头法裤子、皮肤颜色、红色衣服、淡蓝色的鞋子，刚好这几个特征都有被试别到。

Report due Mon 11:59pm to MobileSecurity2014@163.com. You can write in either English or Chinese. Document naming convention: ExperimentDate- ChineseName-StudentID, e.g., 2014-11-24-张三-123456789.

2. 跟 OpenCV 的比较:

   a. 迭代一次:

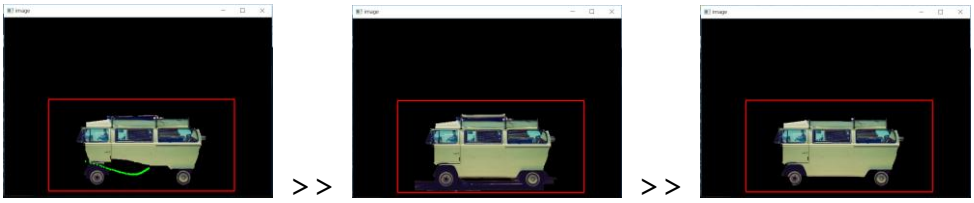
(OpenCV)

(我的)

我的跟 OpenCV 的比较，看起来我的分割效果较好一点，虽然我的前景裤子的部份被认为是背景了，但起码没有把背景的部份误认为前景，而且 OpenCV 分割的鞋子的部份也没有处理好。

还有一点，OpenCV 内建的 Grabcut 速度非常的快，我觉得内建的 Graph 定义可能有些不一样，导致在做 Flow 的时后结果可能也不一样，或是有省下一些东西，OpenCV 为了追求速度而损失一些准确性，也是合理的。

(1) 前景背景相似、且复杂的场景:



进行了五次迭代，最后两次结果很相近，应该是收殮了，可以看到效果不是那么好，原因有几个就是前景跟背景的颜色实在太像了，在做前景跟背景的 GMM 有一部份是重叠的导致很难准确的分离前景跟背景，还有一个很大的原因是我们的 GMM 的 K 调 5，代表说把整张图片分 10 类，前背景个半，但是这张图片的复杂程度远大于 10 类，因此难以有好的结果。

五、参考文献

Grab Cut : http://alex-phd.blogspot.com/2014/03/graph-cutgrab-cut.html

Max-flow min-cut : http://www.csie.ntnu.edu.tw/~u91029/Cut.html