北京理工大学计算机学院
《数字图像处理》课程报告

题目：Interactive Graph Cuts 的实现

姓名：徐程

学号：1120141839

班级：07111403

联系方式：15624951839

指导教师：梁玮

2017 年 2 月 24 日

Interactive Graph Cuts 的实现

摘　要

本文首先对论文[1]和论文[2]的主要思想进行了较为深入的介绍，然后进一步地将论文[1]提出的 Interactive Graph Cuts 算法进行了代码实现并对实现结果进行了分析和总结。

关键词： Interactive Graph Cuts

目　录

第 1 章　背景

1.1 定义

Interactive Graph Cuts（交互式图像分割）一种利用已知的用户对图像的部分前景和背景进行的标记，将整个图像的前景和背景进行区分的图像处理技术。

1.2 主要思想

论文[1]中用到的能量函数模型如下：

$$\mathrm{E(A)} = \lambda \mathrm{R(A)} + B(A) \tag{1}$$

E(A)为能量函数，其中 R(A)是区域项，B(A)为边界项。

1.2.1 区域项

$$\mathrm{R(A)} = \sum_{p \in P} R_p(A_p) \tag{2}$$

R(A)代表给一个像素点 p 进行标记的代价，也就是说对 p 进行标记的正确率越高，R(A)越小。

论文[1]中给出的计算 R(A)的公式如下：

$$R_p(\text{"obj"}) = -\ln P_r(I_p|O) \tag{3}$$

$$R_p(\text{"}bkg\text{"}) = -\ln P_r(I_p|B) \tag{4}$$

公式(3)、(4)分别代表像素点 p 被标记为前景、背景的代价。

1.2.2 边界项

$$\mathrm{B(A)} = \sum_{\{p,q\} \in N} B_{\{p,q\}} \cdot \delta(A_p, A_q) \tag{5}$$

B(A)为边界项，代表相邻像素点 p、q 的不连续性的代价，p、q 越相似，B(A)项就越大。

论文[1]中给出的计算$B_{[p,q}$的公式如下：

$$B_{\{p,q\}} \propto \exp\left(-\frac{(I_p-I_q)^2}{2\sigma^2}\right)\cdot\frac{1}{dist(p,q)} \quad (6)$$

由于在计算时采用四邻域，所以 dist(p,q)恒等于 1，公式可简化为

$$B_{\{p,q\}} \propto \exp\left(-\frac{(I_p-I_q)^2}{2\sigma^2}\right) \quad (7)$$

Graph Cuts 的目标是最小化 E(A)，所以需要给像素 p 进行最有可能的标记，并且寻找不同标记的相邻像素。

1.2.3 实现步骤

论文[1]提出的 Graph Cuts 技术的实现主要可分为以下三个步骤：

i. 用户对图像进行标记，得到部分已知为前景的像素点和已知为背景的像素点。Graph Cuts 的主要过程就是比较未知像素点与已知为前景或背景像素点的相似度大小来判断并对未知像素点进行标记的过程。对于未标记的像素，若它与已标记为前景的像素点的相似度比已标记为背景的像素点的相似度大，那么它将被标记为前景，反之则被标记为背景。

ii. 根据论文[2],可将每个像素点看作图的一个顶点进行构图，并额外添加两个点：source(源点)以及 sink(汇点)，将这两个点统称为 terminal，即为图的终点，source 代表前景，sink 代表背景，从源点出发连接图中其他顶点的有向边和从图中其他顶点触发连接汇点的有向边称为 t-link,而图中其他顶点间相连的有向边成为 n-link。这样就形成了一个有向图。

iii. 通过特定的规则为有向图中的各有向边赋予权值，然后将这个带权有向图作为输入，利用网络流算法计算出图的最大流，也就是最小割，由此就可以将图中的顶点分为两部分，一部分连接源点，这些点全部表示前景，另一部分连接汇点，这些点全部表示背景。

第 2 章　程序实现

2.1　程序的运行环境、运行步骤

（1）开发环境：Visual studio 2013+ opencv-3.0.0+Windows 7 旗舰版 64 位

（2）注意事项：需要在运行机上安装 opencv。程序运行前需确保活动解决方案的属性->链接器->附加依赖项中包含 opencv_ts300d.lib 和 opencv_world300d.lib；另外由于本计算机为 64 位，所以活动解决方案的活动平台配置为 x64,相应链接器->目标计算机选定 MachineX64(/MACHINE:X64)。

（3）运行步骤：若要用程序自带的图像，则在已安装 Visual Studio 的电脑上点击 ImgTest.sln 直接运行即可，输出结果在 Output 文件夹中，可看到输出的已经分割好的前景和背景；若需要更换别的图像进行验证，则需要在 input 文件夹中放入原图、已经人工标记好的前景图和背景图，再将 main.cpp 中如下图语句中的文件名依次调整为新的图片的原图名，前景图名和背景图名，之后再点击运行即可。

2.2　程序说明

2.2.1 程序结构

本程序主要分为四个部分：

1) 图像分割：ImageSegmentation.cpp, ImageSegmentation.h
2) 最大流算法实现：max_flow.cpp,max_flow.h
3) 主程序：main.cpp
4) 其他均用于图像处理工作：Image.cpp，Image.h，point.h，ImageException.h，lodepng.h，lodepng.cpp,其中 lodepng.h，lodepng.cpp 是用于 c/c++的 PNG 格式图片编码解码库

2.2.2 实现细节

本程序实现了大作业要求的论文[1]提出的 Graph Cuts 算法。

- 最大流算法

  最大流算法的实现采用了 Ford-Fulkerson 算法，其实现思想是利用 BFS（广度优先搜索）寻找增广路径，然后找到路径上的残余流量的最小值 f，并对每条边正向减去最小值 f,每条边反方向的流量加 f，重复以上步骤同时将每条增广路径的 f 值累加直到找不到下一条增广路径为止，也就是说，此时从图的源点 source 无法到达汇点 sink 了，此时便可求出该图的最大流（f 值累加的结果）。

  正如论文[2]所说，利用 BFS 进行广搜需要扫描大量的像素，尤其在输入图像较大的时候是十分耗费时间的，所以论文[2]提出了基于增广路径的新算法，这种算法主要有两点不同，第一点是建立了两个搜索树，一个从源点开始搜索，另一个从汇点开始搜索；第二点是这种方法重用了搜索树而不是重新进行构建。但考虑到这种方法有些难以实现，本程序还是采用了 BFS 算法进行搜索，并且在实现时使用了较小的图像。

- n-link

```
// 相邻两点的边权
int ImageSegmentation::get_weight(int x1, int y1, int x2, int y2) {
    double temp = 0.0;
    double dist = 0.0;
    for (int c = 0; c < image.channels(); ++c)
    {
        temp +=  (image(x1, y1, c) - image(x2, y2, c)) * (image(x1, y1, c) - image(x2, y2, c));
    }
    return (int)(beta *exp(-0.25*temp / (2 * sigma*sigma)));

}
```

图 2-1 程序截图

返回的权值是公式(7)的具体实现，其中数字 0.25 是经过反复实验后确定的使分割效果比较好的比例系数,sigma 表明相邻元素之间的不相似度，由 getSigma( ) 函数计算。

- t-link

```
// 添加连接源点和汇点的边
for (int x = 0; x < width; ++x) {
    for (int y = 0; y < height; ++y) {
        int pt1 = x + y * width;
        double fg_prior = getPr(x, y, 1);
        double bg_prior = getPr(x, y, 2);

        mf->add_edge(mf->s(), pt1, (int)(lambda * fg_prior), (int)(lambda * fg_prior));
        mf->add_edge(pt1, mf->t(), (int)(lambda * bg_prior), (int)(lambda * bg_prior));
```

图 2-2 程序截图

其中添加的边的权值是公式（3）、（4）的具体实现，其中 lambda 表示区域项 R(A)的重要程度,经过多次实验最终确定了默认最佳数值数值为 0.4。

2.3 程序运行结果

经过实验，我认为有两个直接影响实验结果的因素，一个是区域项所占比重 lambda,另一个是边界项 n-link 权值公式中的比例系数 beta，下面是从数次实验中挑出的较具有代表性的不同因素取不同值对应的实验结果：

原始图片为：

图 2-3 原始图片

1） 取 lambda=1.0,beta=1.0:

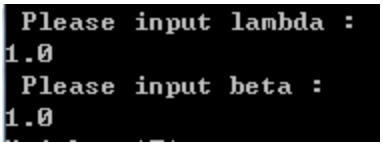

图 2-4 输入数据

实验结果如下：

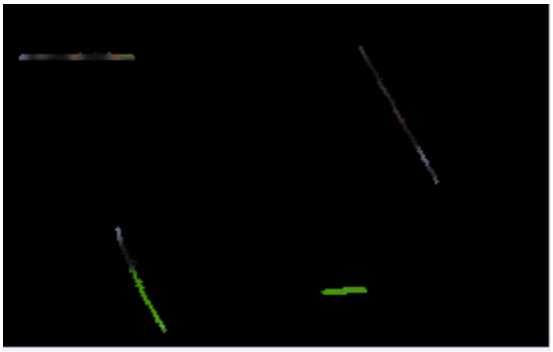

图 2-5 背景

图 2-6 前景

2） 取 lambda=0.2,beta=50.0:

```
 Please input lambda :
0.2
 Please input beta :
50.0
```

图 2-7 输入数据

实验结果如下：

图 2-8 背景

图 2-9 前景

3) 取 lambda=0.4,beta=1000.0：

```
 Please input lambda :
0.4
 Please input beta :
1000.0
```

图 2-10 输入数据

实验结果如下：

图 2-11 背景

图 2-12 前景

第 3 章　总结和感悟

从 2.3 程序运行结果可以看出，实验三次取值的图像分割效果呈递增关系，一次较一次好，其中最后一次 lambda, beta 取值是目前实验效果中最佳的，然而也可以从第三次实验结果中看出还有白色的一部分背景和图中人物脚下的一小片草地都被划分为前景，说明本程序还有许多不足的地方。总的来说，这次大作业在实现的过程中遇到了许多困难，首先由于布置的两篇论文是英文的，在研读和理解上就耗费了较多的精力和时间，尤其是论文[1]；边界项和区域项计算公式的确定是本次大作业一大难点和关键点所在；另外在实现最大流算法的时候有向图的构建也较为困难，可以说，Graph Cuts 算法的计算量也是比较大的。

经过本次作业，我深刻地意识到自己在对算法的理解能力和实现能力上严重的不足，并且图像分割技术只是图像处理领域较为基础的技术。经过本学期梁老师对图像处理课程的基础知识进行了认真地讲解，使我对图像处理领域有了较为整体的认识，并且掌握了部分较为基础的知识，另外梁老师经常给我们播放的一些视频让我了解图像处理领域一些最新的科技和科研成果，确实激起了我对图像处理的兴趣，我觉得老师的目的不仅是为了要让我掌握课程要求的基础知识，而且还让我对这门课涉及的整个领域有了较为整体的理解，让我明白这个领域的学者都在从事什么样的研究，有哪些新奇有趣的技术，这一点非常重要。

参考文献

[1] Y. Y. Boykov and M. P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images," Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001, Vancouver, BC, 2001, pp. 105-112 vol.1.

[2] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max- flow algorithms for energy minimization in vision," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 9, pp. 1124-1137, Sept. 2004.