第五次上机实验报告

一、 任务介绍

将图像纹理合成问题转换为最大流最小割问题，实现文章 *“Graphcut Textures: Image and Video Synthesis Using Graph Cuts”* 中的图像纹理合成部分。

二、 算法思想及实现

- 图像读写
  使用 opencv，图像为 Mat 类，每一个点的颜色为 RGB 值。
- 图像放置
  用随机数生成重叠区域的宽和高，随机放置新图像。
- 图像切割
  将切割问题转换为最大流最小割问题：设 s 和 t 为图像中的相邻两点，A(s)为原图像在 s 点的颜色，A(t)为原图像在 t 点的颜色，B(s)为新图像在 s 点的颜色，B(t)为新图像在 t 点的颜色，令 s 和 t 之间的匹配代价为 M(s, t,A,B) = ||A(s)−B(s)||+||A(t)−B(t)||，其中||·||在此算法中为 RGB 值的平方和的平方根。将重叠区域的点作为图的结点，将相邻两点间的代价作为边的权重，由于重叠区域边缘上的点需限制为原图像/新图像的颜色，因此源点和汇点连接到边缘点的边的权重为正无穷。需要找到使代价和最小的切割即使得权重之和最小的切割，即为最大流最小割问题。找到切割后，靠近源点的点依旧是原图像的颜色，靠近汇点的点变为新图像的颜色。

三、 实验中问题、理解与解决方法

- 由于图像中点数量多，在处理时注意其坐标值的规律性
- 调用别人写好的文件(maxflow)时注意其接口
- 注意 opencv 的使用

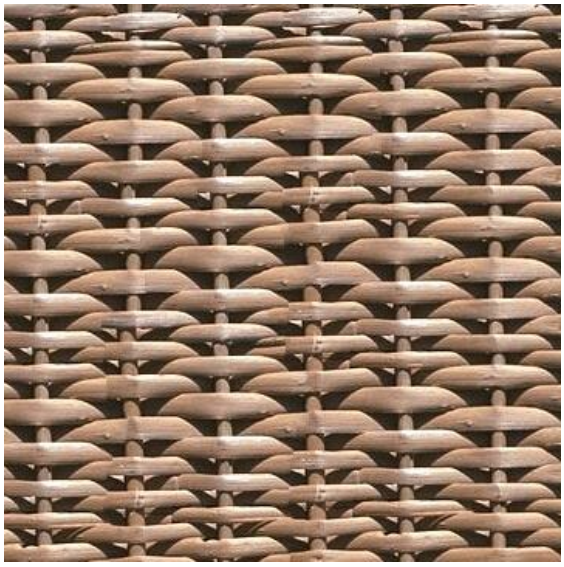
四、 实验结果

先输入待合成的图像文件名：

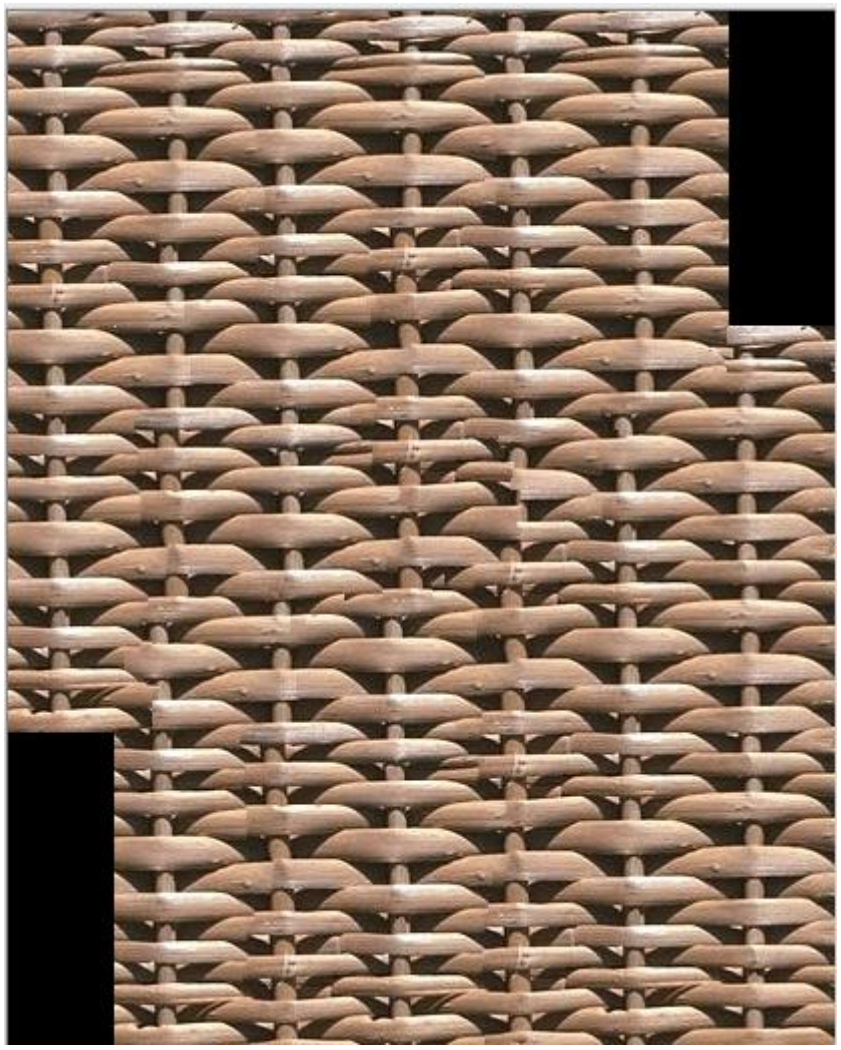
```
choose an image: texture1.png  texture2.jpg  texture3.jpg  texture4.jpg
texture1.png
```

之后进行图像合成（合成两次），以下为四个示例：

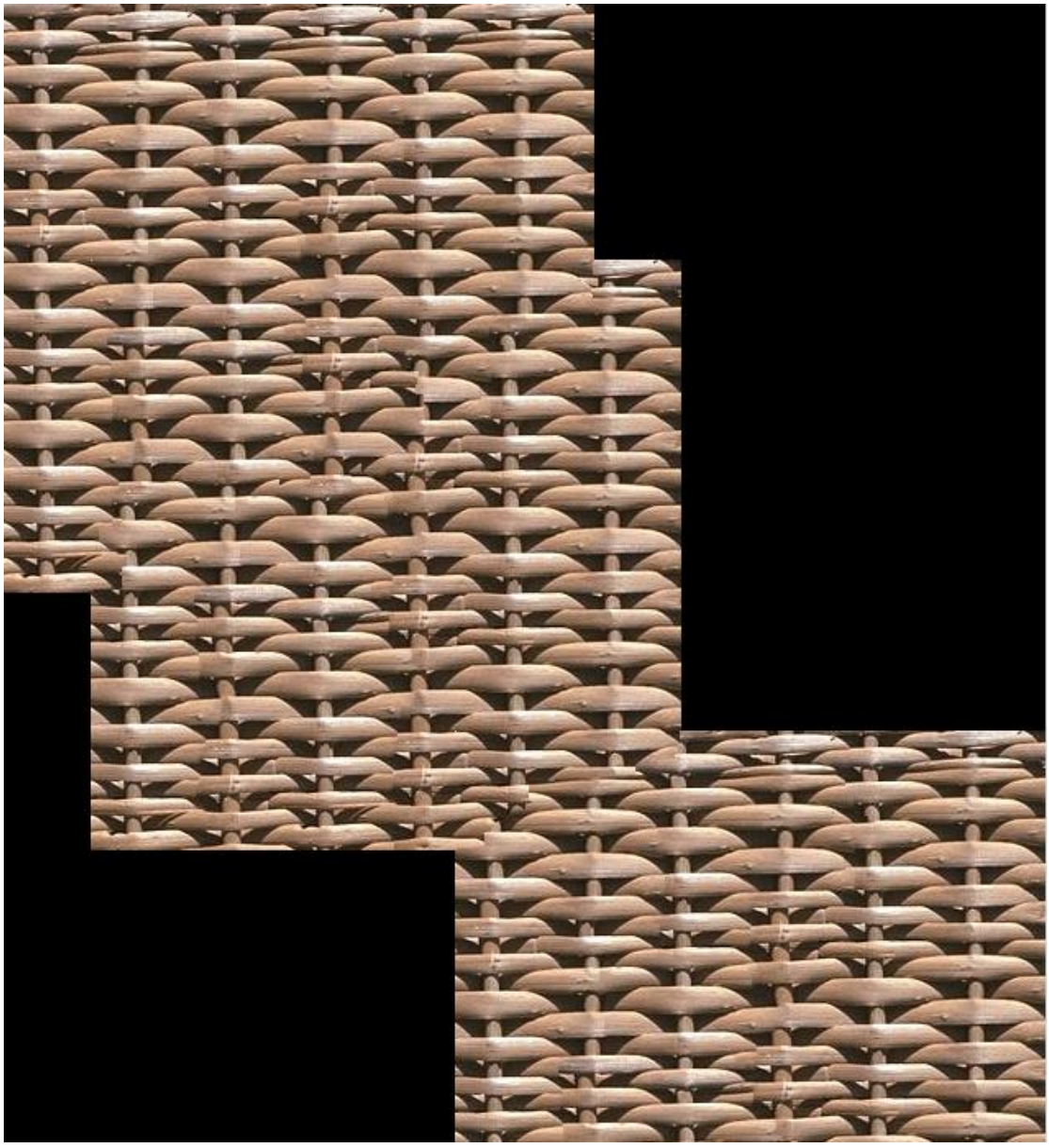
原图：



合成一次：



合成两次：



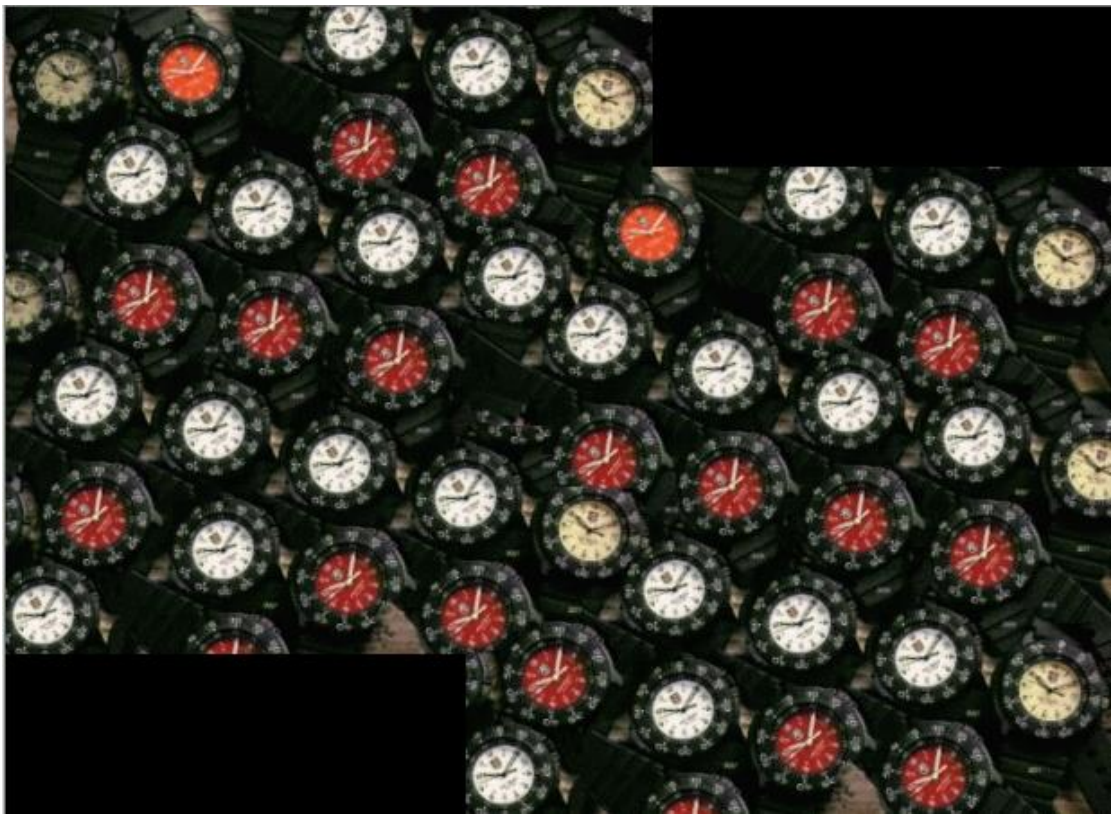
原图：

合成一次：

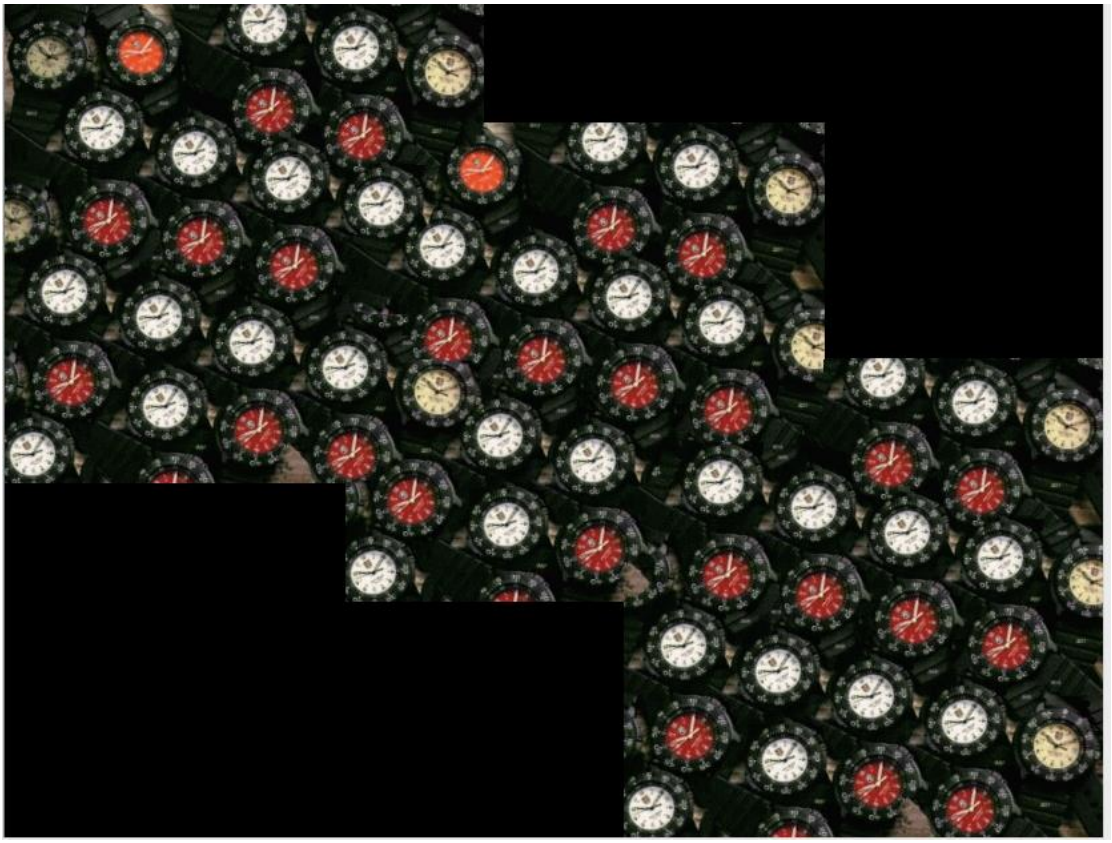
合成两次：

原图：

合成一次：



合成两次：



原图：

合成一次：

合成两次：

从以上示例可看出合成效果良好。