数字图像处理 第 4 次大作业

自 45 柳荫 2014011858

题目：

实现分水岭图像分割算法。发挥想象力，展示算法的中间过程。

解题思路：

涨水，连通域多了相安无事；

少了退回去，膨胀，连通域不少相安无事，少了退回去，修水坝；

保证连通域只多不少。

核心算法及实现：

程序中外层 for 循环中变量 **c** 是核心，每次迭代，无论是仅涨水，还是先修水坝再涨水，c 都是最终得到的图像的集水盆地连通域的逻辑图；同样的，内层循环，变量 **e** 是核心, e 是膨胀到连通域减少前的最后一次膨胀的结果，是集水盆地连通域膨胀若干次的逻辑图。

1， 对一幅 uint8 的灰度图，首先找出其最低“海拔”，则这一个或若干个最低点就是集水盆地最初的小水洼，逐渐开始往外扩散；

2， 用了 matlab 的 bwlabel 函数能够求出一幅二值图各个连通域的分布(本次实验采用了其默认参数，即 8-连通域)，开始“涨水”，最高涨到 255（循环末尾让水涨 1 像素值，所以外层 for 循环中止值是 254）；

3， 每涨一个像素值（变量 b） ，可能连通域变多（出现了一个新的水洼），这时更新连通域的计数值（连通域由 con 计数）；

4， 也可能连通域变少（若干个集水盆地连起来了），这时退回到之前的情况，进入内循环，进行灰度形态学膨胀，逐次膨胀 (变量 d) 直到连通域减少，此时退出内循环，找到连通域个数减小前最后一次膨胀的结果（变量 e），把其每个连通域分别取出来求出其外边界，因为 e 的各个连通域再膨胀一次会出现连通域互连的情况，所以其各个域外边界的交不为空，这些不为空的地方也就是“水坝”应该修筑的地方，要阻碍其互连。

程序架构：

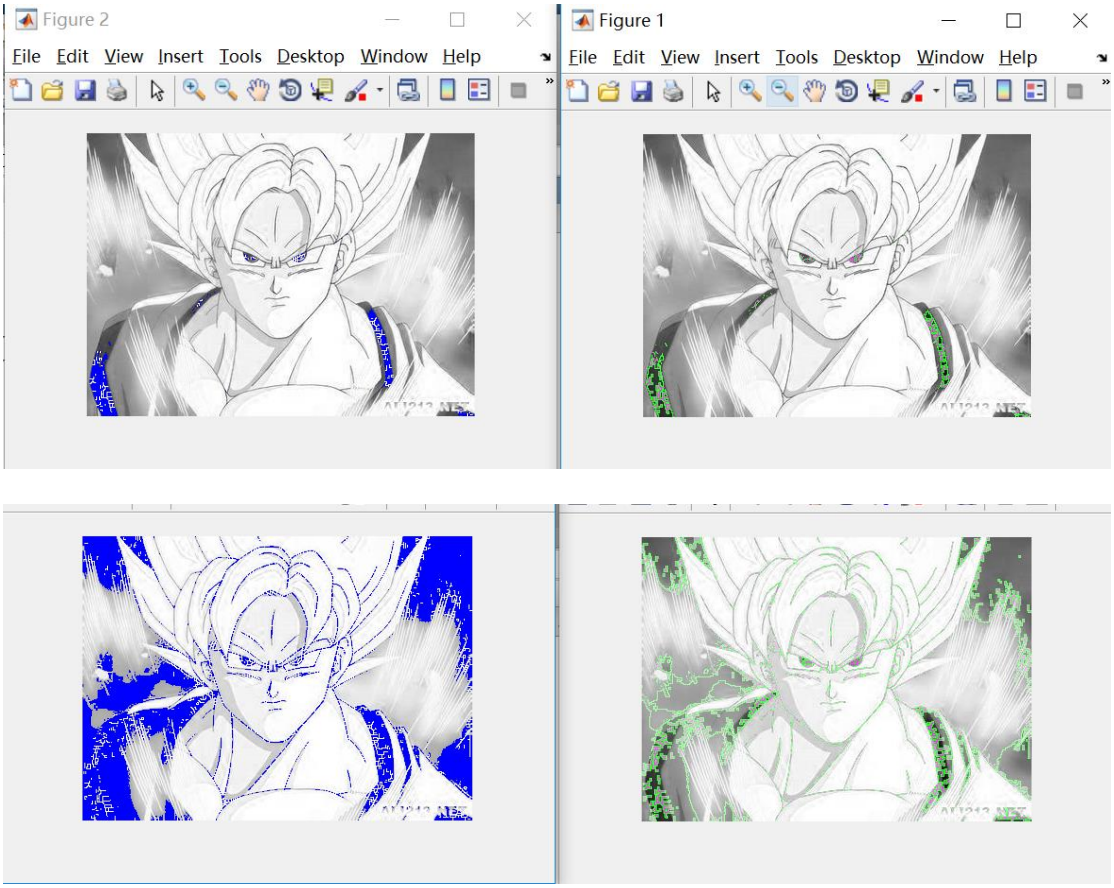
一共编写了一个主脚本 **jbn0.m** 以及 4 个用于显示的函数 **show_process0.m**，**show_process1.m** ，**process2.m** 和 **figure_rgb.m**。其中 **show_process0** 用于显示刚有水洼时的景象，包括连通域的边界；**show_process1** 用于图中的各个连通域，绿色的表示连通域本身自带的边界，紫色的表示要修筑水坝的地方；**process2** 用于在另一张图（figure2）上随着迭代过程动态显示“涨水”的过程，除了实时修建的水坝外，其余部分当被水淹没时，显蓝色，表示已被浸没； **figure_rgb** 与 **show_process1** 基本一样，除了最后水坝的 3 个通道都要加上去，它用于做 3D 显示的效果。

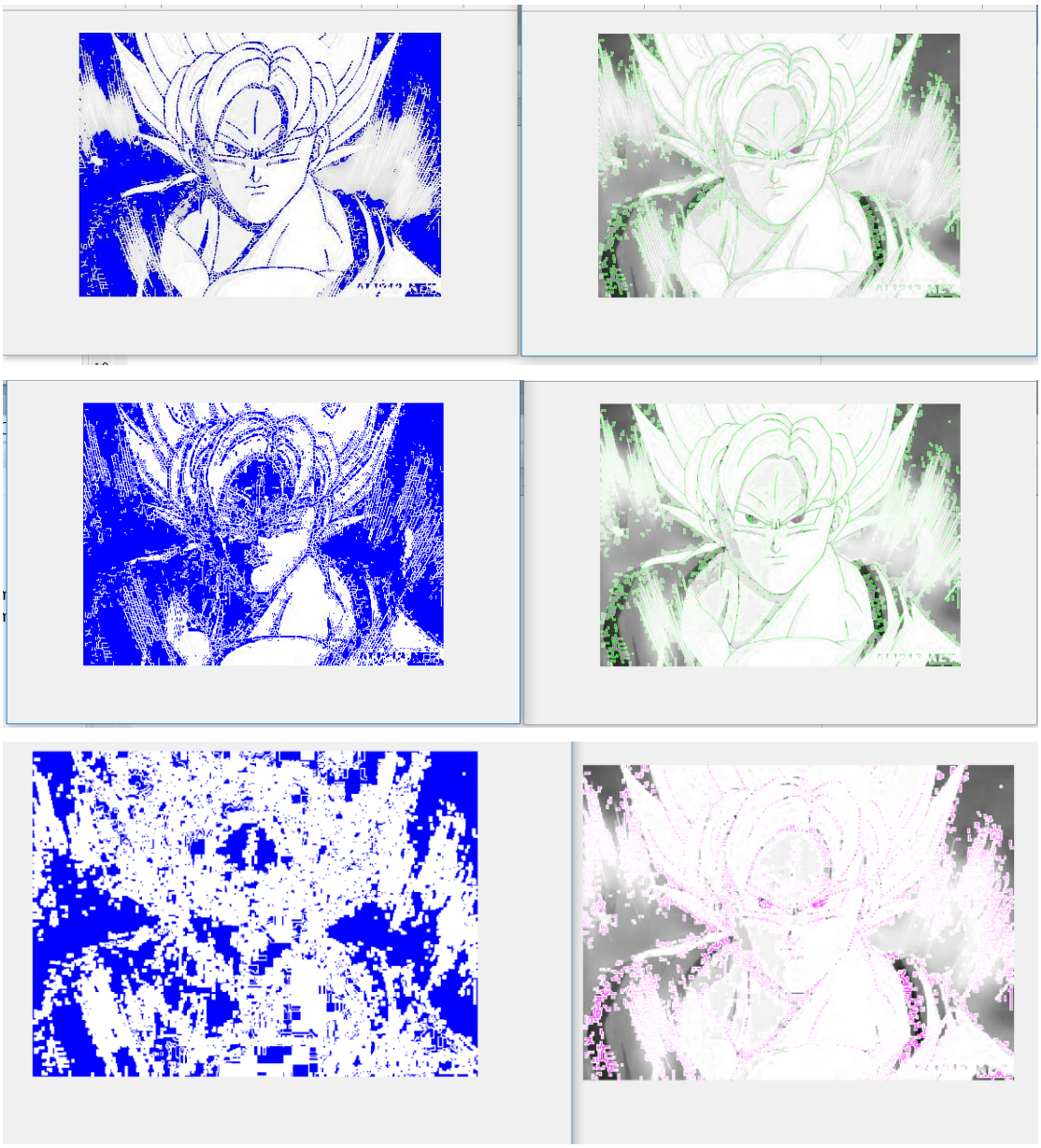
运行效果：

运行函数脚本如下(自行选取了一些有趣的灰度图)：

原图： 

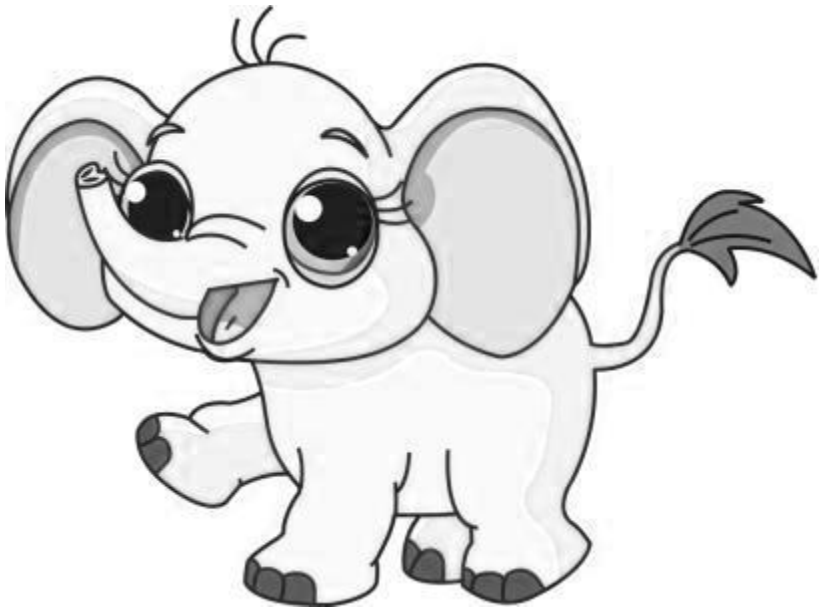
处理中：





前面几组是动态涨水和修水坝的过程（颜色的含义详见上文），最后一组图片，右侧的 figure1 中，紫色部分表示最终需要新修水坝的地方；左边的 figure2 中，白色的部分表示最后一次迭代后，各个连通域的边界 ***加上*** 修筑的水坝，为什么看上去比右边的紫色部分多那么多？原因就在于此图最终在默认 bwlabel 定义和 ones(3,3)的结构元素的条件下，连通域的个数达到了惊人的 2463 个,于是边界众多，是严重的过分割现象，由于过分割现象的消除应该在预处理阶段进行，因此本次作业分水岭算法的实现暂不考虑。

再一组图片：



原图：

处理中：

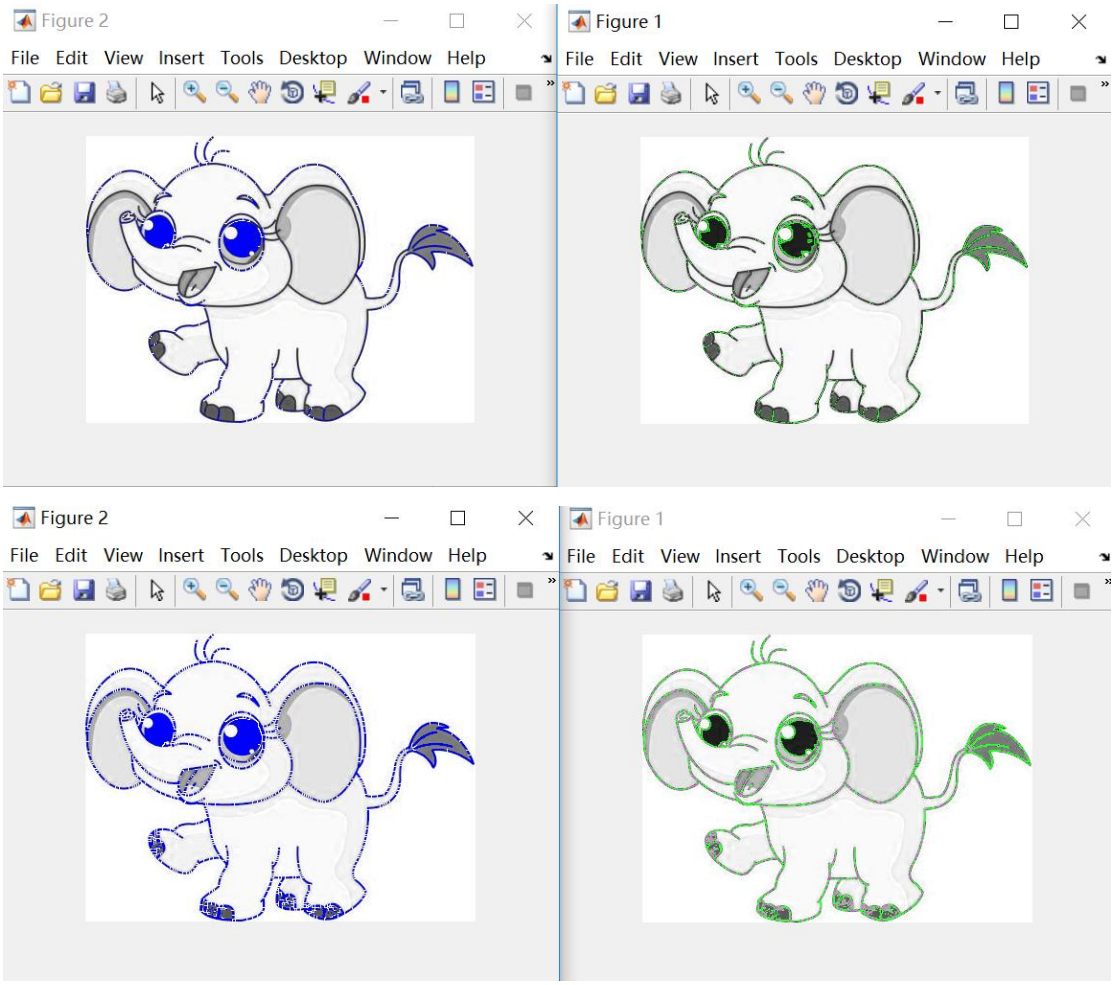


Figure 2
File Edit View Insert Tools Desktop Window Help
Figure 1
File Edit View Insert Tools Desktop Window Help
Figure 2
File Edit View Insert Tools Desktop Window Help
Figure 1
File Edit View Insert Tools Desktop Window Help

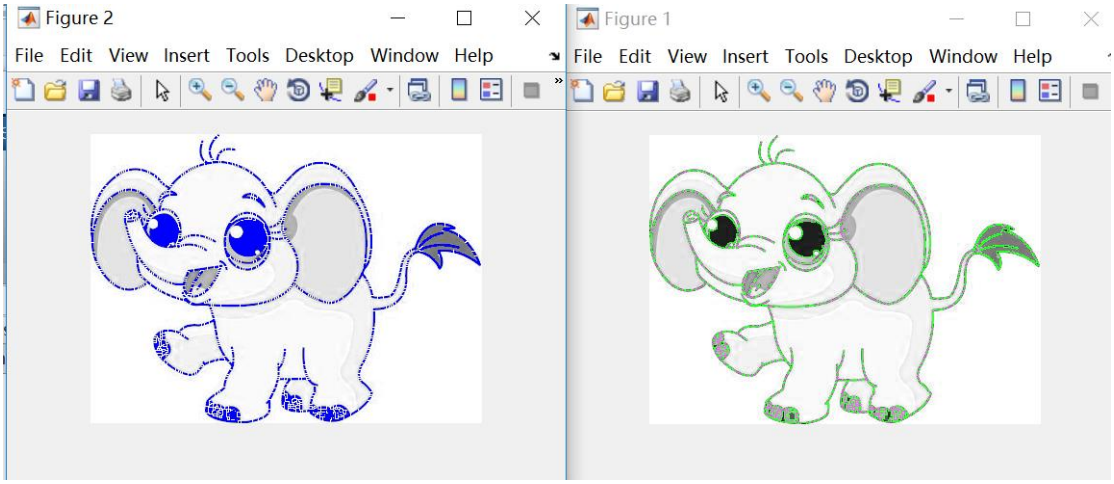


Figure 2
File Edit View Insert Tools Desktop Window Help
Figure 1
File Edit View Insert Tools Desktop Window Help

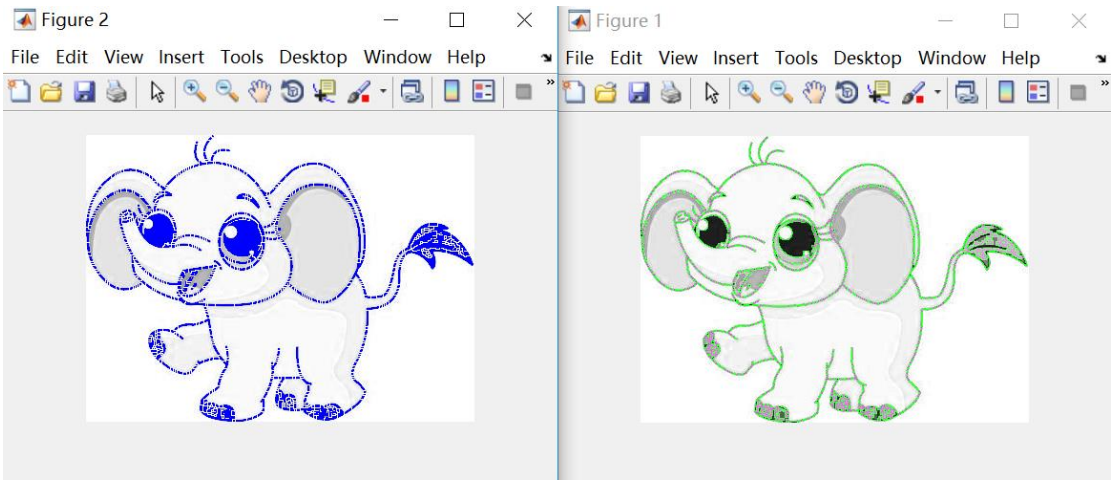
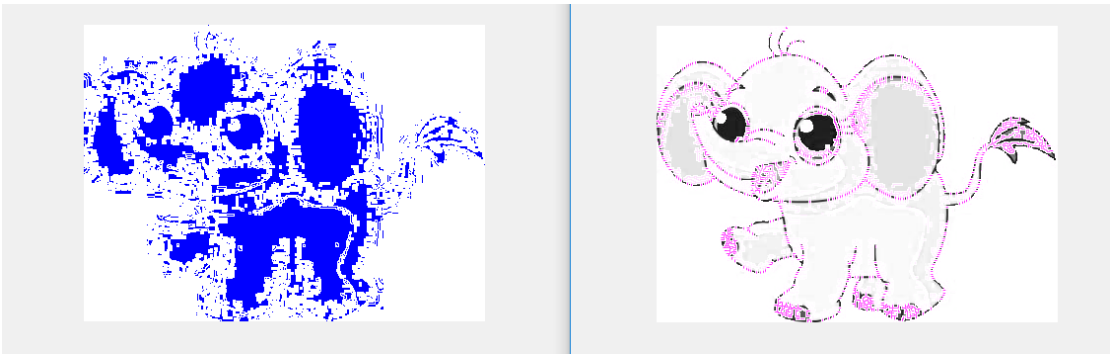
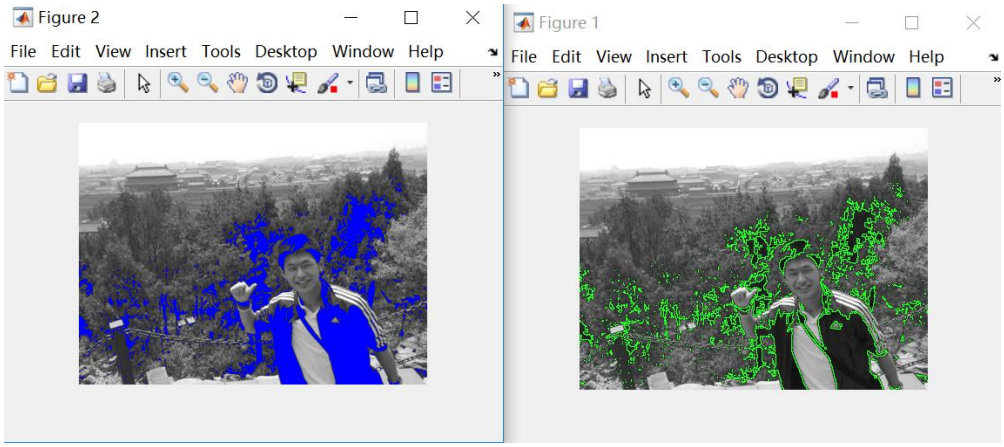


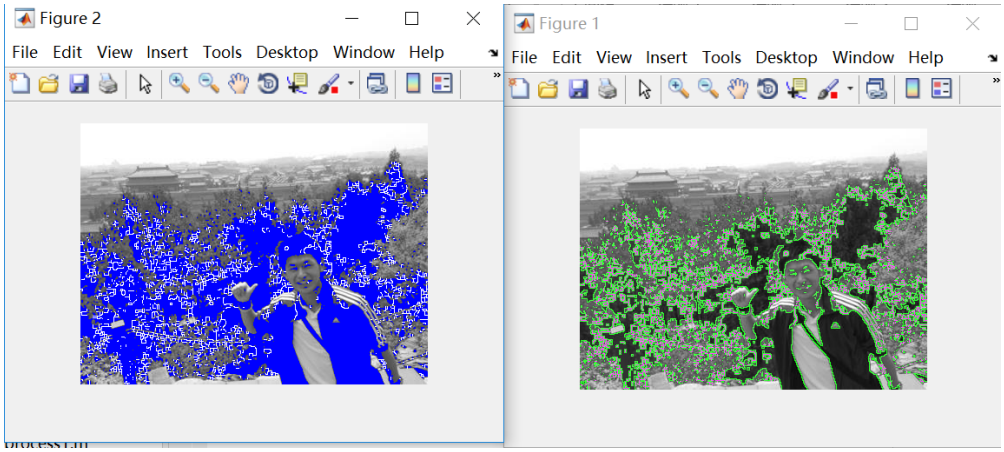
Figure 2
File Edit View Insert Tools Desktop Window Help
Figure 1
File Edit View Insert Tools Desktop Window Help

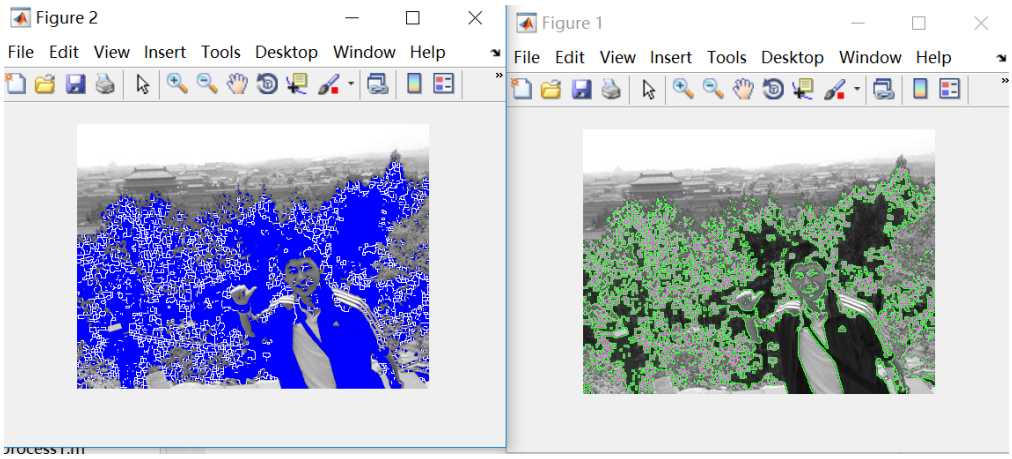


再一组图：

原图：  （我本人的照片转成灰度）







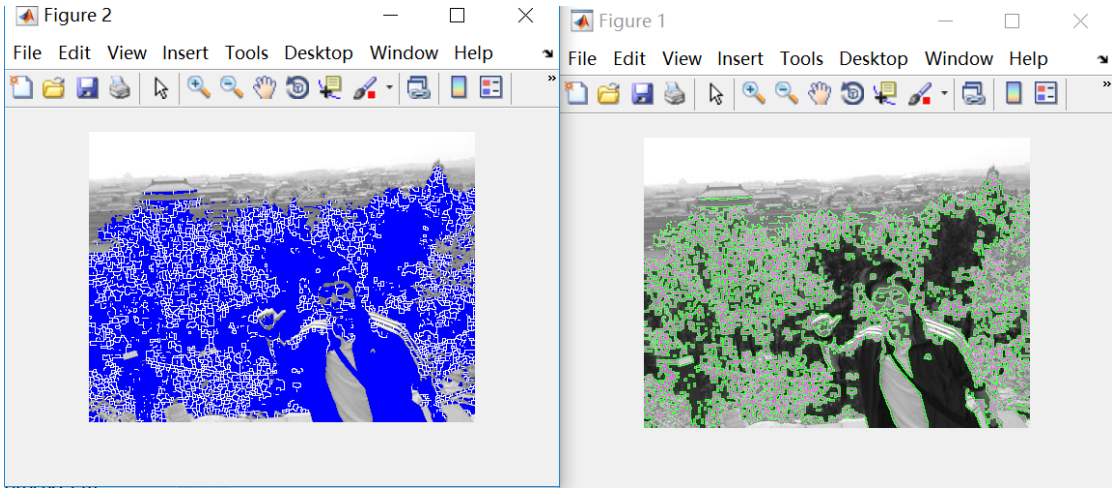


Figure 2
File Edit View Insert Tools Desktop Window Help
Figure 1
File Edit View Insert Tools Desktop Window Help

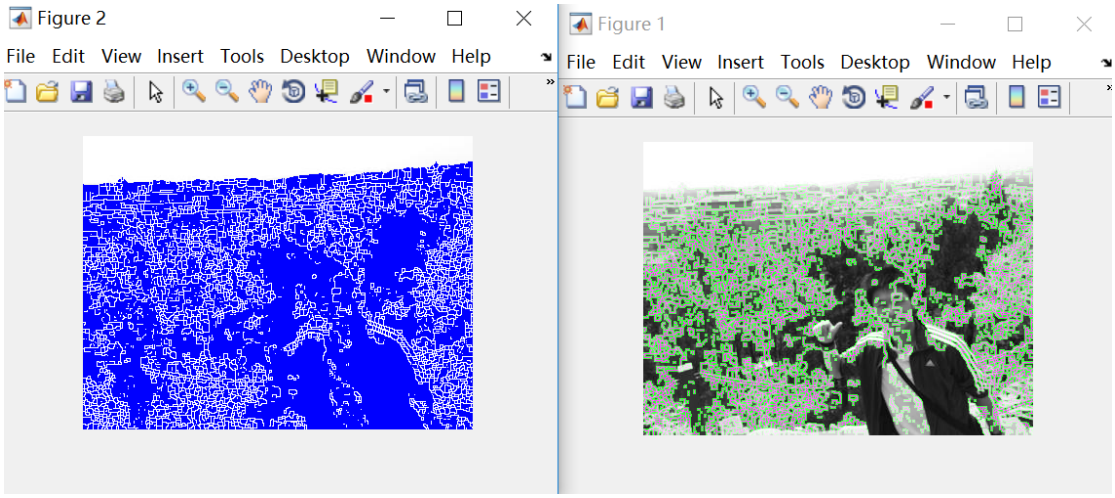
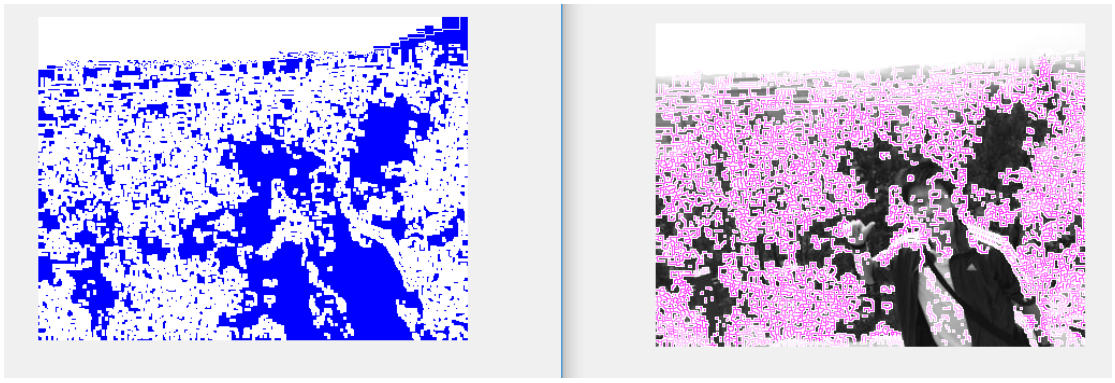
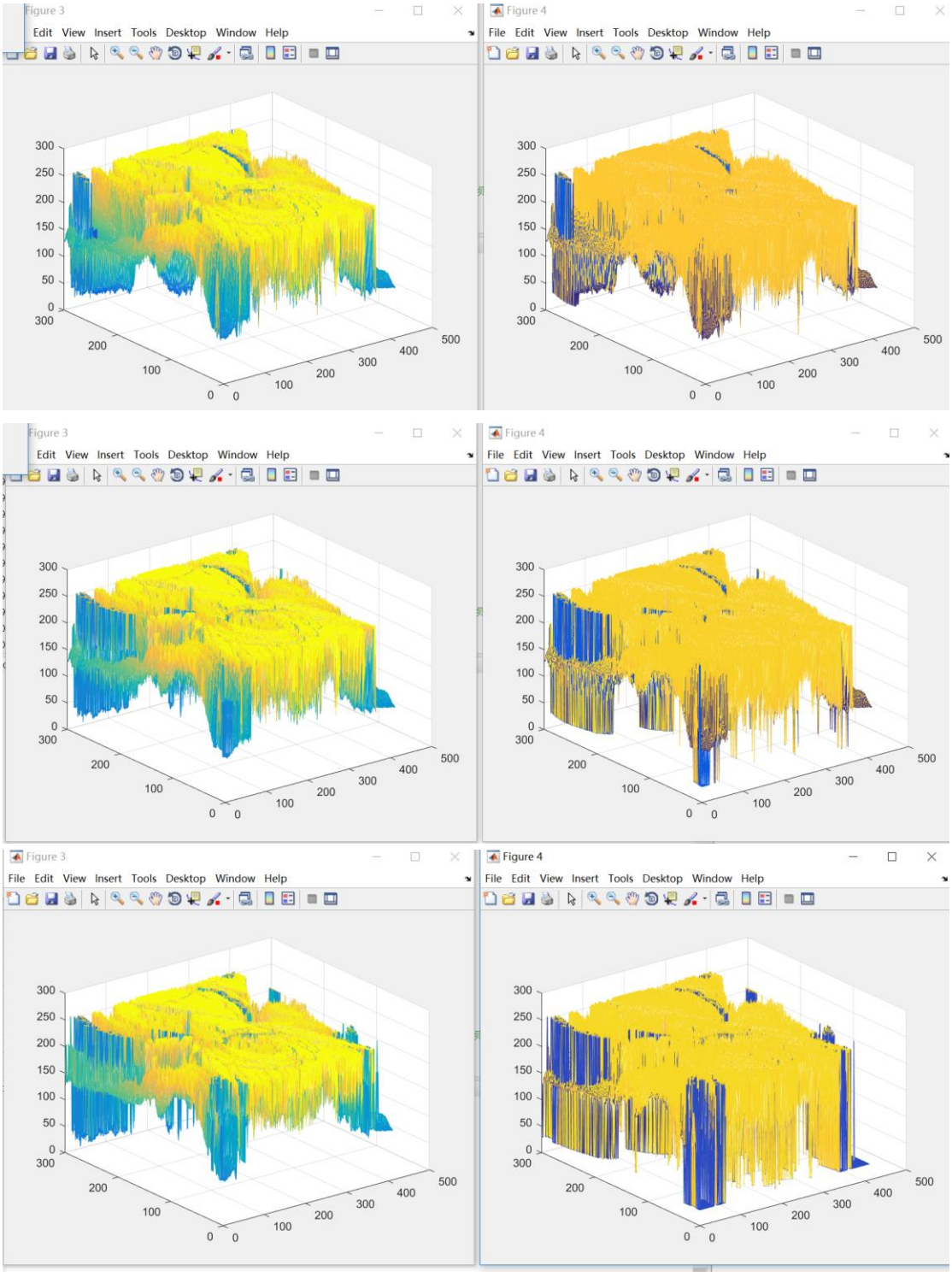


Figure 2
File Edit View Insert Tools Desktop Window Help
Figure 1
File Edit View Insert Tools Desktop Window Help



3 维动态显示：

程序中用了简单的 mesh 函数做了一种 3D 的动态效果，以上面第一张原图为例，其 figure(3),figure(4) 分别如下变化，其中 figure(3)对应于 figure(1)，figure(4)对应于 figure(2)：



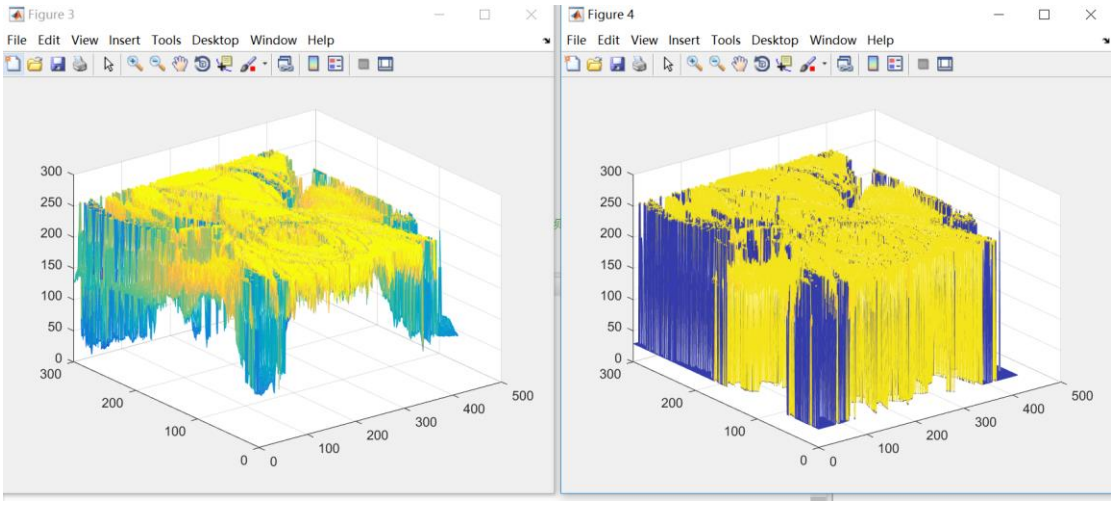


Figure 3
File Edit View Insert Tools Desktop Window Help
300
250
200
150
100
50
0
300
200
100
0
0
100
200
300
400
500

Figure 4
File Edit View Insert Tools Desktop Window Help
300
250
200
150
100
50
0
300
200
100
0
0
100
200
300
400
500

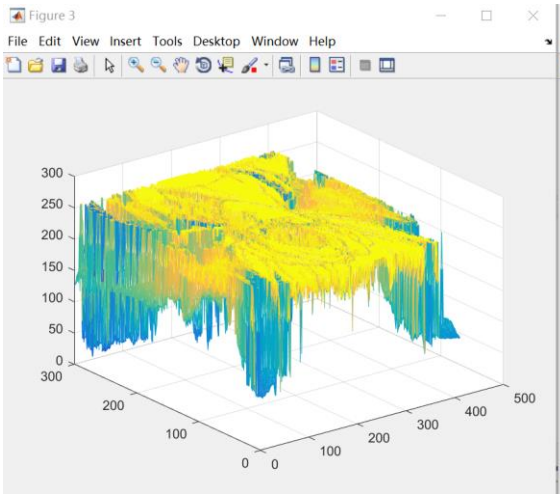


Figure 3
File Edit View Insert Tools Desktop Window Help
300
250
200
150
100
50
0
300
200
100
0
0
100
200
300
400
500

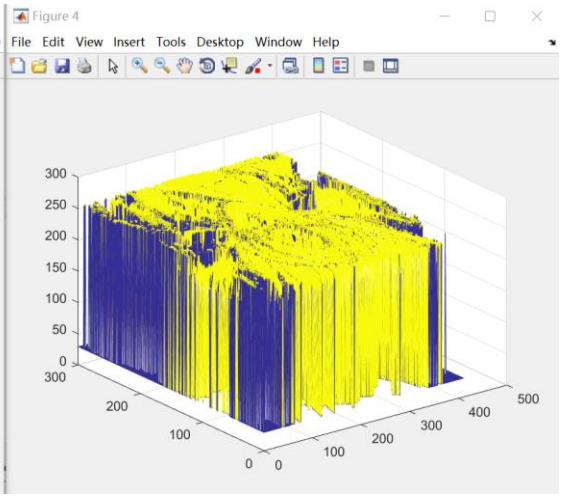


Figure 4
File Edit View Insert Tools Desktop Window Help
300
250
200
150
100
50
0
300
200
100
0
0
100
200
300
400
500

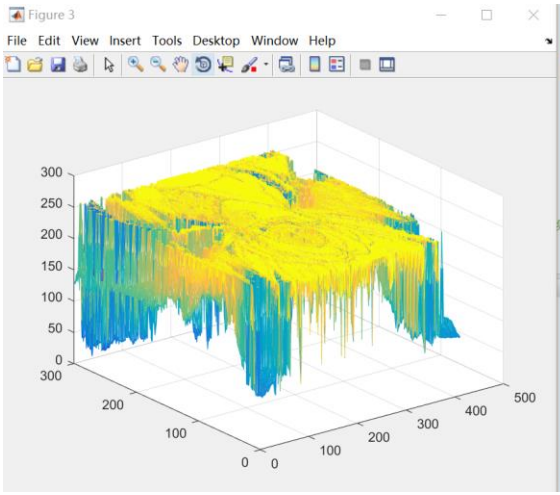


Figure 3
File Edit View Insert Tools Desktop Window Help
300
250
200
150
100
50
0
300
200
100
0
0
100
200
300
400
500

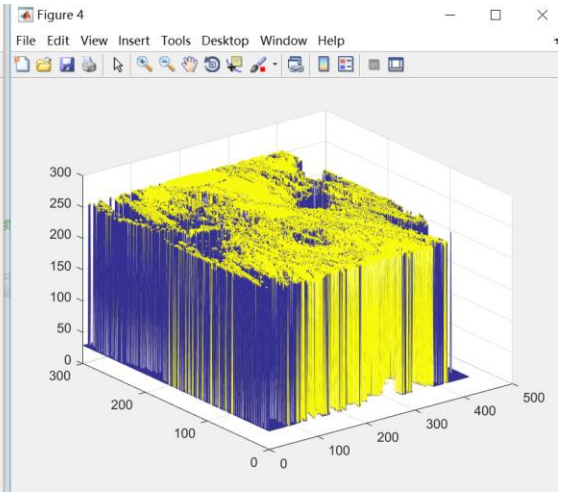


Figure 4
File Edit View Insert Tools Desktop Window Help
300
250
200
150
100
50
0
300
200
100
0
0
100
200
300
400
500

实验感悟：

1，本次大作业，刚开始由于老师说网上代码很多，我就想依赖网上的资源，但不幸的是，没有找到合适的分水岭算法，要么所用语言不熟悉算法不好懂，要么就是实现效果不够好，最终就完全自己动手写，效果本人觉得也还行；

2，分水岭算法的大作业是一次我们自己发挥创造力，进行建模构造编程的过程，这不像从前可以参考不少老师的代码或是直接调用函数，锻炼了本人的matlab 编程技巧，比如摸索中怎么退回上一步，边界什么时候算到水坝中去以及连通域边界、水坝、海水的显示方法等等，都经过了本人的一次次探索尝试而做出来；

3，会了一点基础的 3D 图像显示技巧；

4，巩固了我形态学算法的相关内容。