数字图像处理

第一次作业

姓名：王浩然

班级：自动化 61

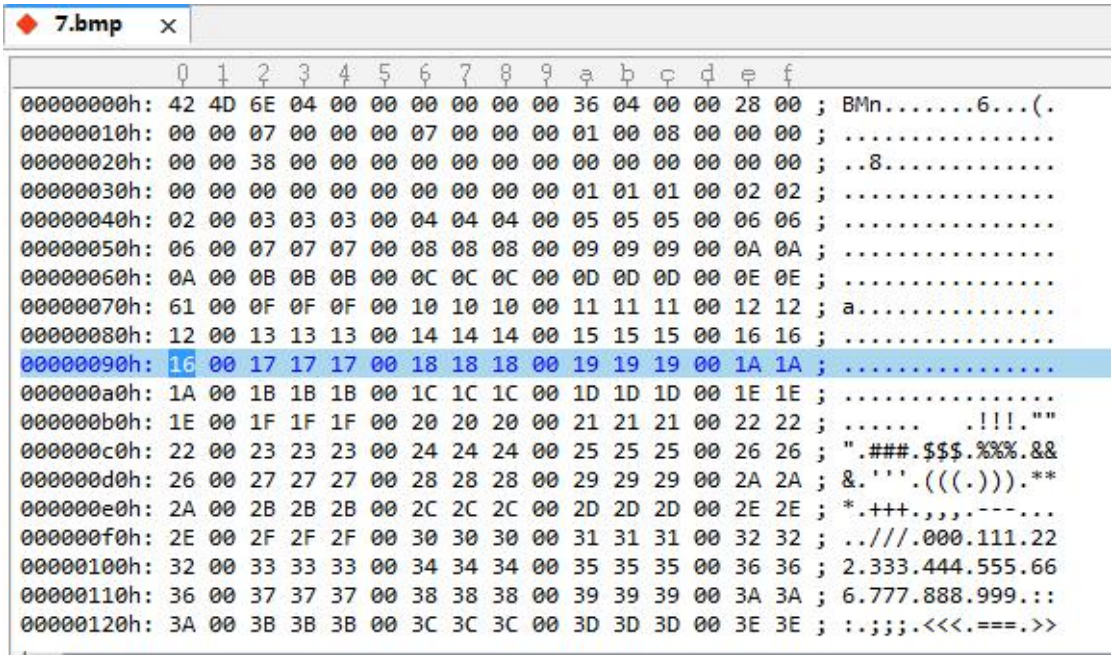
学号：2160504020

提交时间：2019/3/4

摘要：数字图像处理在生活中使用频率极高，对图像格式的理解和图像的处理是必须的，图像的均值以及方差的计算是基础，近邻、双线性和双三次插值法在图像处理中使用频繁，其图像的旋转也是必不可少的一环。本次试验用 C++和 OpenCV 库进行对图片的各种处理，实现数字图像处理功能。

1、Bmp 图像格式简介,以 7.bmp 为例说明；
BMP 文件由文件头、位图信息头、颜色信息和图形数据四部分组成。
~~位图头文件数据结构，它包含 BMP 图像文件的类型、显示内容等信息；
~~位图信息数据结构，它包含有 BMP 图像的宽、高、压缩方法，以及定义颜色等信息；
~~图片的调色板。
~~位图数据，这部分的内容根据 BMP 位图使用的位数不同而不同，在 24 位图中直接使用 RGB，而其他的小于 24 位的使用调色板中颜色索引值。

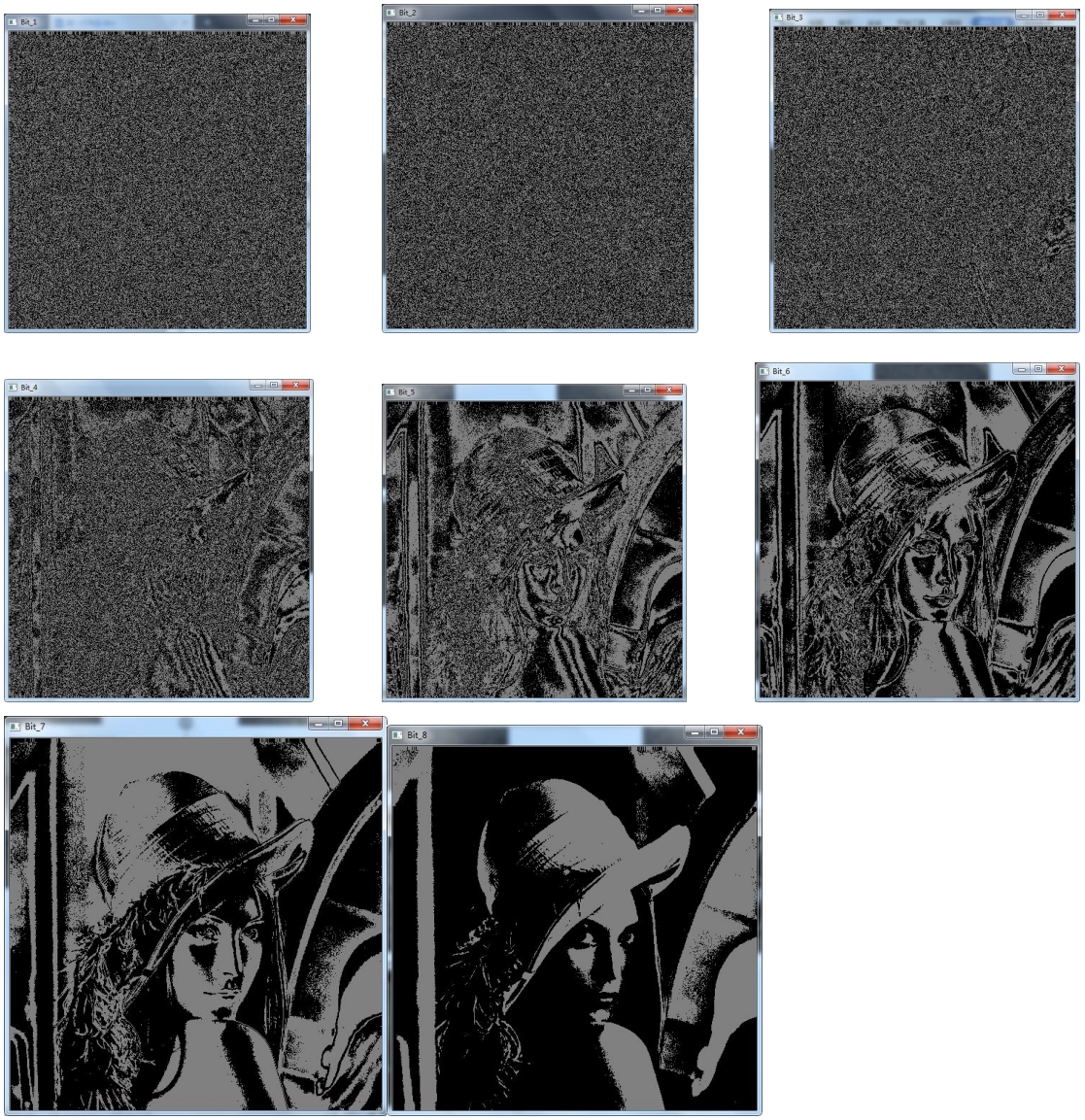
以 7.bmp 为例，7.bmp 的进制格式如下：



其中：
~~BMP 文件头数据结构含有 BMP 文件的类型、文件大小和位图起始位置等信息。
00000000h 中 42 4D 表示文件的类型是 BMP，占两个字节。
00000000h 中 6E 04 00 00 表示文件大小，占四个字节,大小为 1134 个字节。
00000000h 中 abcd 列表示位图数据的起始位置
~~BMP 位图信息头数据用于说明位图的尺寸等信息。
00000010h 中 00 00 00 07 00 00 00 07 表示位图的宽度高度，其中，宽高都为 7 像素。
~~颜色表用于说明位图中的颜色
~~位图数据记录了位图的每一个像素值

2、把 lena 512*512 图像灰度级逐级递减 8-1 显示；

由于灰度图像每个像素是 8BIt，而每 BIT 的信息不同。
灰度级递减即把每一个 Bit 的信息以单独的一个 Bit 以图片的形式显示出来即可。
实验结果：下图为 Bit1----Bit8 的各 Bit 层信息

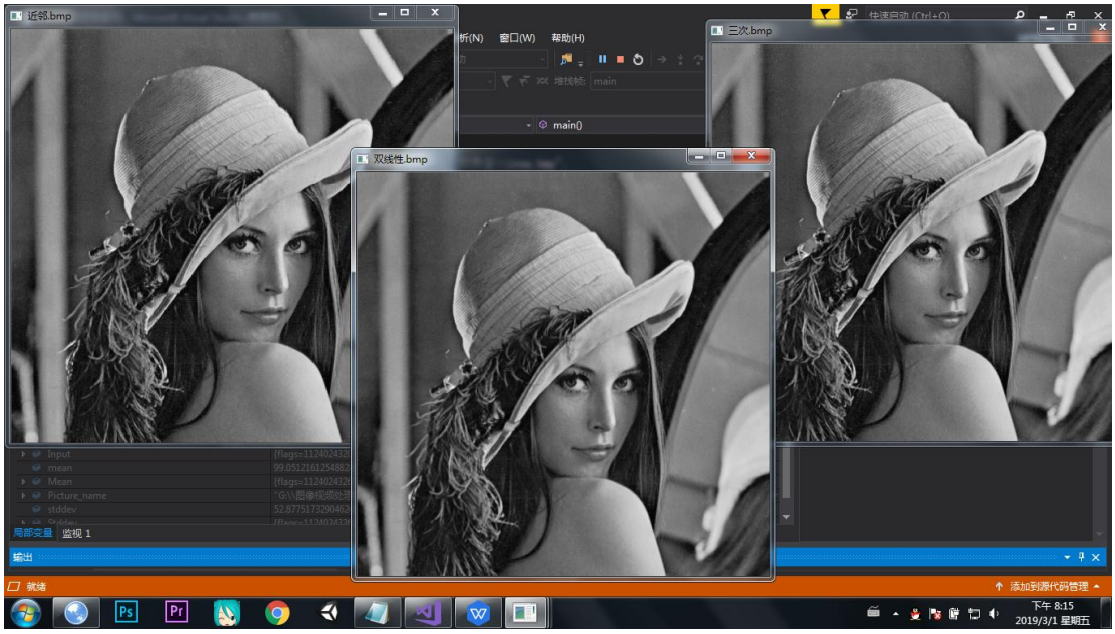


3、计算 lena 图像的均值方差；

图片的均值与方差即灰度的均值与方差，灰度的范围为 0--255
对一个图片的所有灰度进行统计后求均值与方差即可
实验中用到了 OpenCV 库中的 meanStdDev 函数直接求得了输入图像 Input 的均值和方差
实验结果：

4、把 lena 图像用近邻、双线性和双三次插值法 zoom 到 2048*2048；

插值方法在图像处理中十分常用，近邻插值法是最简单的插值方法，把变换后的矩阵中缺少的像素值用 srcX=dstX* (srcWidth/dstWidth) , srcY = dstY * (srcHeight/dstHeight)求出即可，单一般近邻插值法效果不是很好；
双线性插值法是根据于待求点 P 最近 4 个点的像素值，计算出 P 点的像素值，相较于近邻插值法其对像素的相关性好了许多，故图片看起来更自然。
双三次插值法是利用原图像距离像素(x,y)最近的 16 个像素点作为计算目标图像（x,y）处像素值的参数。
具体实现中用到了 OpenCV 中的 resize 函数，resize 函数定义了要输入的处理图像以及处理后的图像，根据所需最终图片大小选择插值方式：CV_INTER_NN 近邻插值，CV_INTER_LINEAR 双线性插值， CV_INTER_CUBIC 双三次插值，来对图像进行 resize。
实验结果：下图从左到右依次为 lena 图像用近邻、双线性和双三次插值法 zoom 到 2048*2048 的结果



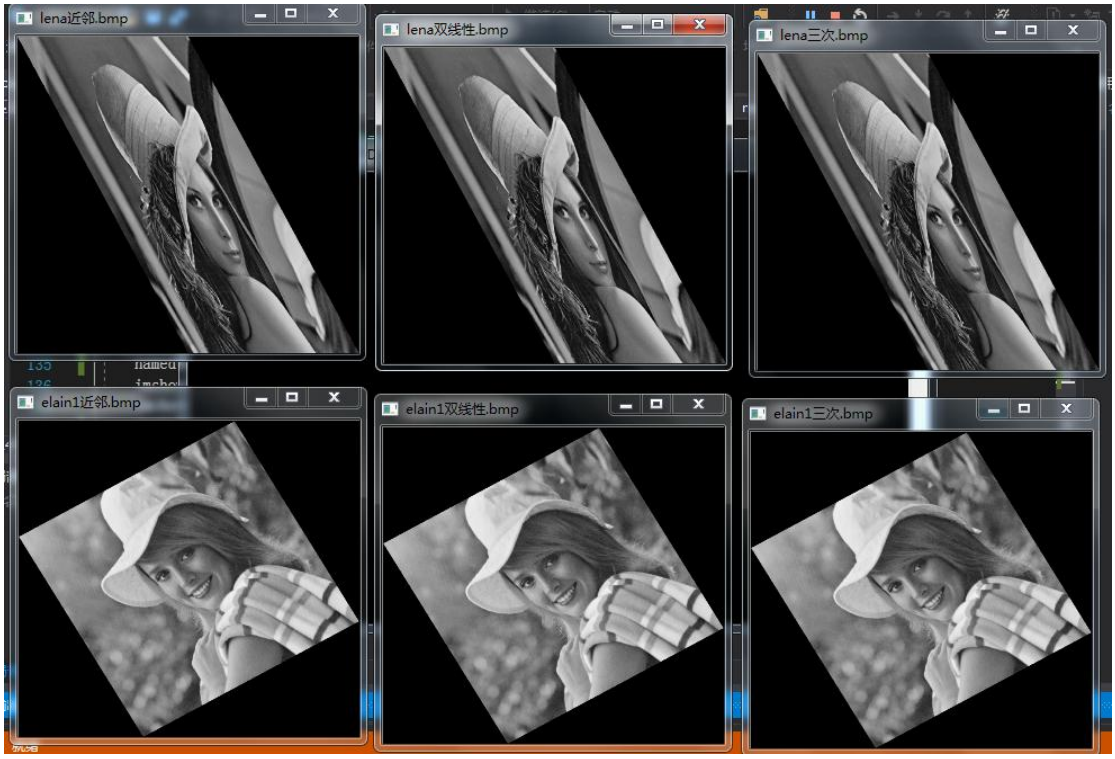
5、把 lena 和 elain 图像分别进行水平 shear（参数可设置为 1.5，或者自行选择）和旋转 30 度，并采用用近邻、双线性和双三次插值法 zoom 到 2048*2048；

这一步与上一题很像，只要是多了一步水平 shear 和旋转
水平 shear 其实就和第二次作很像了，把变换钱和变换后的点 Point2f 定以后，用 getAffineTransform 和 warpAffine 进行变换就可得到水平 shear 后的图片。
旋转 30 度即对图像进行仿射变换，图像绕原点逆时针旋转 a 角，其变换矩阵及逆矩阵（顺时针选择）为

$$M=\begin{bmatrix}\cos\alpha & \sin\alpha & 0\\ -\sin\alpha & \cos\alpha & 0\end{bmatrix}\quad M^{-1}=\begin{bmatrix}\cos(-\alpha) & \sin(-\alpha) & 0\\ -\sin(-\alpha) & \cos(-\alpha) & 0\end{bmatrix}=\begin{bmatrix}\cos\alpha & -\sin\alpha & 0\\ \sin\alpha & \cos\alpha & 0\end{bmatrix}$$

实现过程中用 RotatedRect 输入旋转图片和角度，以及旋转中心，然后用得到的参数进行 warpAffine 即可达到相应的图片；
得到旋转和水平 shear 后得图片后在井陉第四题中的操作即可

实验结果：



参考文献

1.Adrian Kaehler and Gary Bradski，Learning OpenCV 3， 1005 Gravenstein Highway North, Sebastopol, CA 95472，2017

2.阮秋琦 等，数字图像处理（第三版），电子工业出版社，ISBN：9787121313837，2017