

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、双线性插值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:

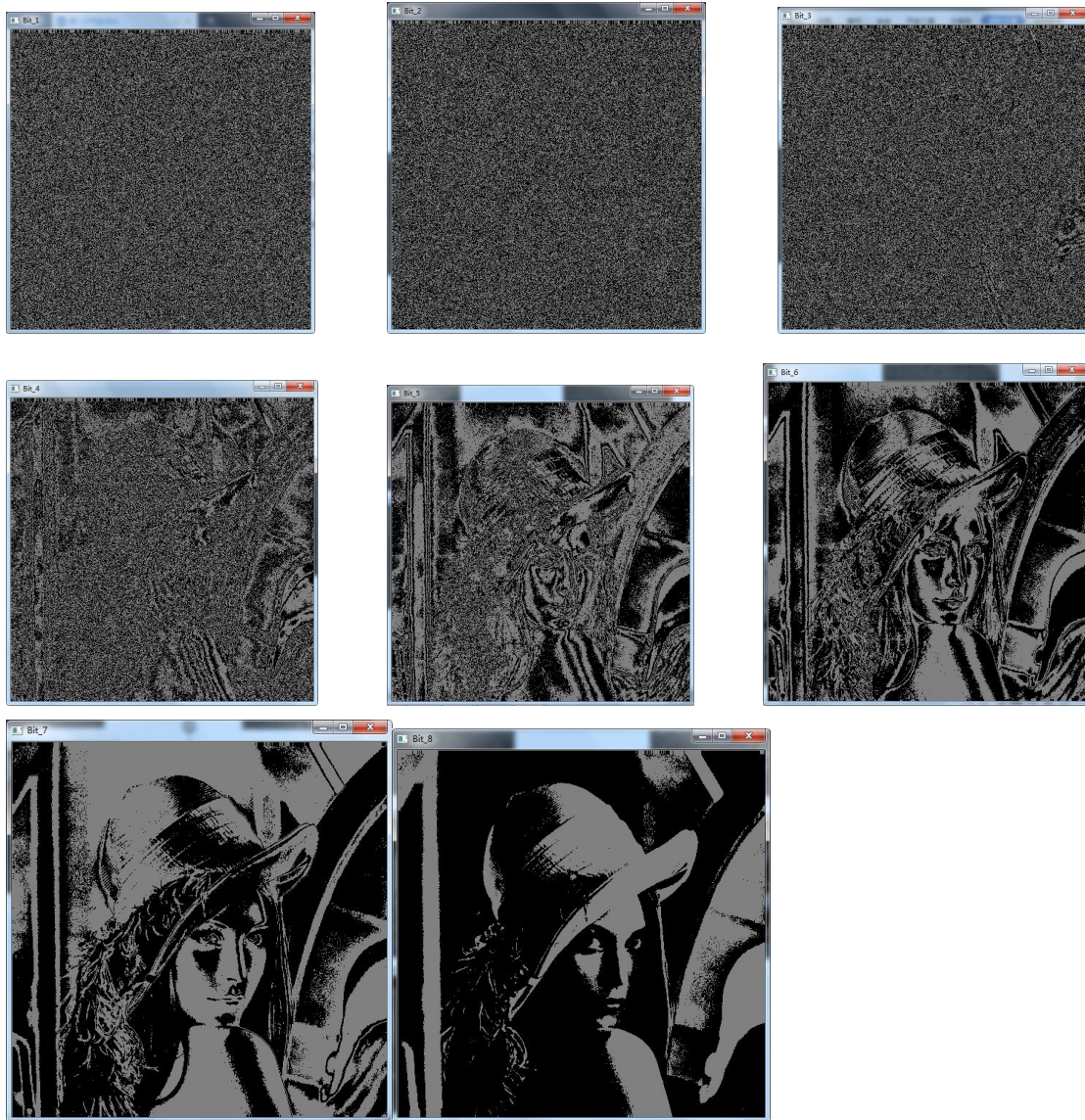
7.bmp x

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f																																																																																																																																																																																																																																																																																																																					
00000000h:	42	4D	6E	04	00	00	00	00	00	00	36	04	00	00	28	00	; BMn.....6...(. 00000010h:	00	00	07	00	00	00	07	00	00	00	01	00	08	00	00	00	; 00000020h:	00	00	38	00	00	00	00	00	00	00	00	00	00	00	00	00	00	; ..8..... 00000030h:	00	00	00	00	00	00	00	00	00	00	00	01	01	01	00	02	02	; 00000040h:	02	00	03	03	03	00	04	04	04	00	05	05	05	00	06	06	; 00000050h:	06	00	07	07	07	00	08	08	08	00	09	09	09	00	0A	0A	; 00000060h:	0A	00	0B	0B	0B	00	0C	0C	0C	00	0D	0D	0D	00	0E	0E	; 00000070h:	61	00	0F	0F	0F	00	10	10	10	00	11	11	11	00	12	12	; a..... 00000080h:	12	00	13	13	13	00	14	14	14	00	15	15	15	00	16	16	; 00000090h:	16	00	17	17	17	00	18	18	18	00	19	19	19	00	1A	1A	; 000000a0h:	1A	00	1B	1B	1B	00	1C	1C	1C	00	1D	1D	1D	00	1E	1E	; 000000b0h:	1E	00	1F	1F	1F	00	20	20	20	00	21	21	21	00	22	22	; !!!."" 000000c0h:	22	00	23	23	23	00	24	24	24	00	25	25	25	00	26	26	; ".###.\$\$\$.%%.&& 000000d0h:	26	00	27	27	27	00	28	28	28	00	29	29	29	00	2A	2A	; &.''.(((.)).** 000000e0h:	2A	00	2B	2B	2B	00	2C	2C	2C	00	2D	2D	2D	00	2E	2E	; *.+++.,,.-.-... 000000f0h:	2E	00	2F	2F	2F	00	30	30	30	00	31	31	31	00	32	32	; ..///.000.111.22 00000100h:	32	00	33	33	33	00	34	34	34	00	35	35	35	00	36	36	; 2.333.444.555.66 00000110h:	36	00	37	37	37	00	38	38	38	00	39	39	39	00	3A	3A	; 6.777.888.999.: 00000120h:	3A	00	3B	3B	3B	00	3C	3C	3C	00	3D	3D	3D	00	3E	3E	; :.;.;.<<<==>>

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：

```
E:\VS_C++\Course\x64\Debug\Course.exe
G:\图像视频处理\第一次作业\第二次作业\lena.bmp的灰度均值是: 99.0512
G:\图像视频处理\第一次作业\第二次作业\lena.bmp的标准差是: 52.8775
```


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

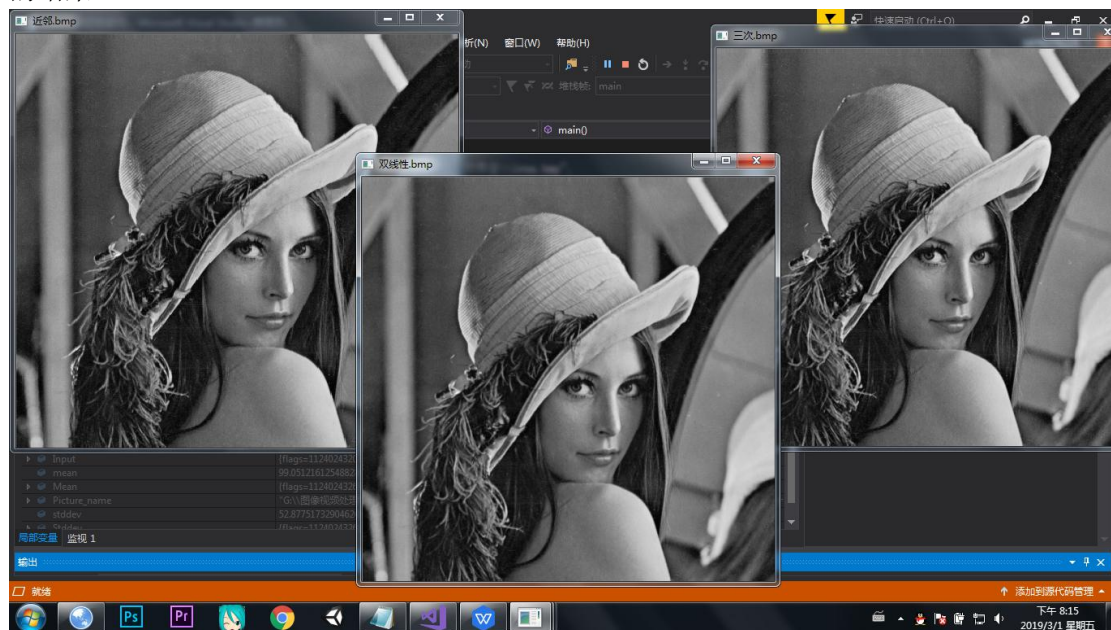
插值方法在图像处理中十分常用，近邻插值法是最简单的插值方法，把变换后的矩阵中缺少的像素值用 $srcX=dstX*(srcWidth/dstWidth)$, $srcY = dstY * (srcHeight/dstHeight)$ 求出即可，单一般近邻插值法效果不是很好；

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。

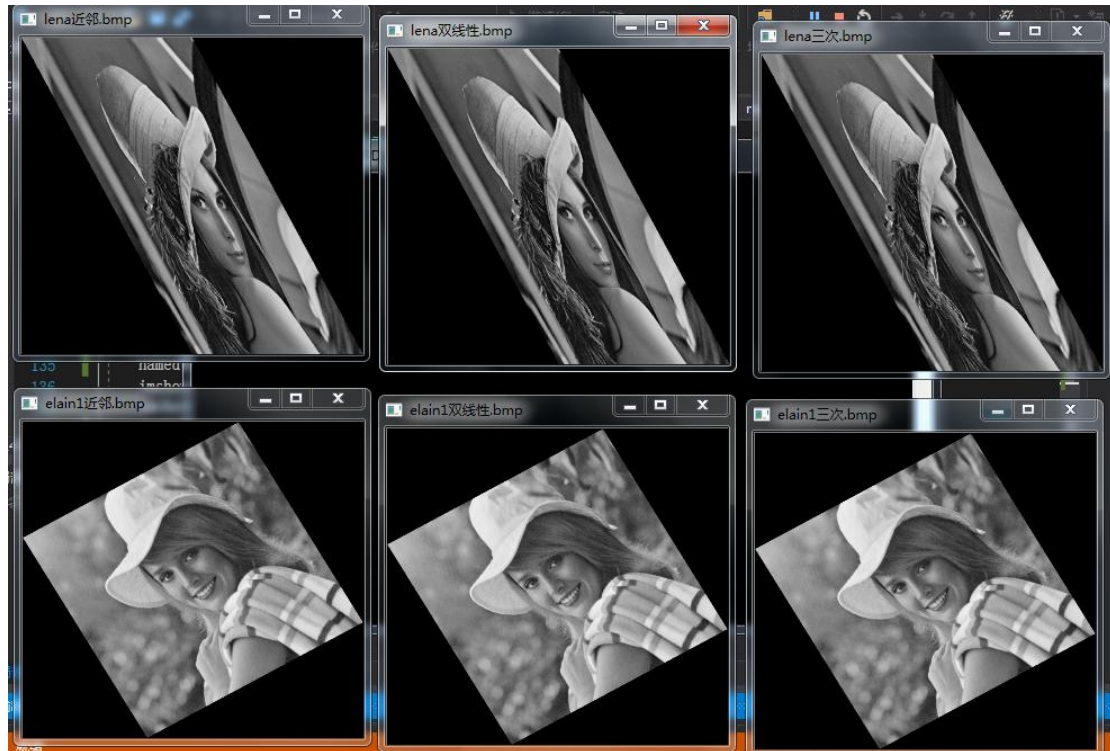
旋转 30 度即对图像进行仿射变换，图像绕原点逆时针旋转 α 角，其变换矩阵及逆矩阵（顺时针选择）为

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

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 后得图片后在并陞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