项目名称：数字图像处理第一次作业

摘要：BMP 位图文件是 Windows 系统下最常用的图形文件格式，所以理解 BMP 文件的内部结构是十分有必要的。图像为了适应各种不同的需要，就必须进行各种各样的变换，其中比较典型的变换有放大缩小，剪切旋转变化得到。其中在对图像进行放大变换的时候，由于图像像素点的增多，因此新图像上的每个点的灰度值需要进行内插操作，而内插又有多种多样的方式。在对图像进行见=剪切旋转等变换时，旋转矩阵的书写也是至关重要的，它决定着新图像的形状。

关键字：BMP 文件详细结构　内插方法　变换矩阵

姓名：金瑞达

班级：自动化 61

学号：2160504013

提交日期：2019.3.3

一．BMP 图像格式简介（以 7.bmp 为例说明）

BMP 文件格式，又称为 Bitmap（位图）或是 DIB(Device-Independent Device，设备无关位图)，是 Windows 系统中广泛使用的图像文件格式。由于它可以不作任何变换地保存图像像素域的数据，因此成为我们取得 RAW 数据的重要来源。Windows 的图形用户界面（graphical user interfaces）也在它的内建图像子系统 GDI 中对 BMP 格式提供了支持。[1]

下面以 EditPlus 为分析工具，以 7.bmp 为例，结合 Windows 的位图数据结构对 BMP 文件格式进行一个深度的剖析。

1.1 BMP 文件数据

BMP 文件的数据按照从文件头开始的先后顺序分为四个部分：

①BMP 文件头(BMP file header)：提供文件的格式、大小等信息。

②位图信息头(bitmap information)：提供图像数据的尺寸、位平面数、压缩方式、颜色索引等信息。

③调色板(color palette)：可选，如使用索引来表示图像，调色板就是索引与其对应的颜色的映射表。

④位图数据(bitmap data)：就是图像数据。

具体如表 1 所示。

表 1 BMP 文件各部分信息

数据段名称	大小（byte）	开始地址	结束地址
位图文件头(bitmap-file header)	14	0000h	000Dh
位图信息头(bitmap-information header)	40	000Eh	0035h
调色板(color table)	由 biBitCount 决定	0036h	未知
图片点阵数据(bitmap data)	由图片大小和颜色定	未知	未知

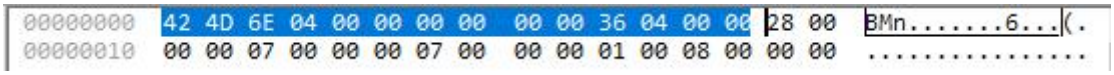
1.2 BMP 文件头

表 2-1 BMP 文件头数据

变量名	地址偏移	大小	作用说明
bfType	0000h	2Bytes	文件标识符，必须为"BM"，即 0x424D 才是 Windows 位图文件 'BM'：Windows 3.1x, 95, NT,… 'BA'：OS/2 Bitmap Array 'CI'：OS/2 Color Icon 'CP'：OS/2 Color Pointer 'IC'：OS/2 Icon 'PT'：OS/2 Pointer 因为 OS/2 系统并没有被普及开，所以在编程时，你只需判断第一个标识"BM"就行

bfSize	0002h	4Bytes	整个 BMP 文件的大小（以位 B 为单位）
bfReserved1	0006h	2Bytes	保留，必须设置为 0
bfReserved2	0008h	2Bytes	保留，必须设置为 0
bfOffBits	000Ah	4Bytes	说明从文件头 0000h 开始到图像像素数据的字节偏移量（以字节 Bytes 为单位），以为位图的调色板长度根据位图格式不同而变化，可以用这个偏移量快速从文件中读取图像数据

打开文件 7.bmp，大小为 1134byte，根据小端模式存储的特点，首先对文件头（0000h-000Dh）进行解析，数据如图 1 所示，解析如表 2-2 所示。



图一 BMP 文件文件头信息

地址	变量名	数值（HEX）	说明
0000h-0001h	bfType	4D42h	0x424D 才是 Windows 位图文件
0002h-0005h	bfSize	046Eh	整个 BMP 文件的大小(1134byte)
0006h-0009h	bfReserved1 和 bfReserved2	00000000h	保留
000Ah-000Dh	bfOffBits	00000436h	字节偏移量（以字节 Bytes 为单位）

表 2-2 文件头信息解析

1.3 BMP 信息头

表 3-1 BMP 信息头数据

变量名	地址偏移	大小	作用说明
biSize	000Eh	4Bytes	BNP 信息头即 BMP_INFOHEADER 结构体所需要的字节数（以字节为单位）
biWidth	0012h	4Bytes	说明图像的宽度（以像素为单位）
biHeight	0016h	4Bytes	说明图像的高度（以像素为单位）。这个值还有一个用处，指明图像是正向的位图还是倒向的位图，该值是正数说明图像是倒向的即图像存储是由下到上；该值是负数说明图像是倒向的即图像存储是由上到下。大多数 BMP 位图是倒向的位图，所以此值是正值。
biPlanes	001Ah	2Bytes	为目标设备说明位面数，其值总设置为 1
biBitCount	001Ch	2Bytes	说明一个像素点占几位（以比特位/像素位单位），其值可为 1,4,8,16,24 或 32
biCompression	001Eh	4Bytes	说明图像数据的压缩类型，取值范围为： 0 BI_RGB 不压缩（最常用） 1 BI_RLE8 8 比特游程编码（BLE），只

			用于 8 位位图 2　　BI_RLE4 4 比特游程编码（BLE），只用于 4 位位图 3　　BI_BITFIELDS 比特域（BLE），只用于 16/32 位位图 4
biSizeImage	0022h	4Bytes	说明图像的大小，以字节为单位。当用 BI_RGB 格式时，总设置为 0
biXPelsPerMeter	0026h	4Bytes	说明水平分辨率，用像素/米表示，有符号整数
biYPelsPerMeter	002Ah	4Bytes	说明垂直分辨率，用像素/米表示，有符号整数
biClrUsed	002Eh	4Bytes	说明位图实际使用的调色板索引数，0：使用所有的调色板索引
biClrImportant	0032h	4Bytes	说明对图像显示有重要影响的颜色索引的数目，如果是 0，表示都重要。

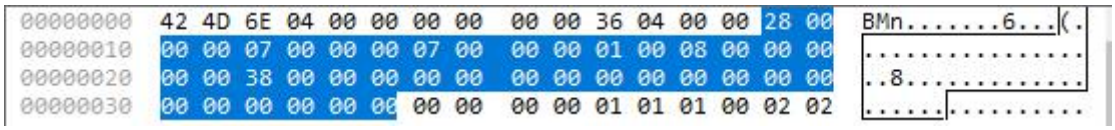


图 3 BMP 文件信息头信息

地址	变量名	数值	说明
0012h-0015h	biWidth	00000007h	宽为 7 像素
0016h-0019h	biHeight	00000007h	高为 7 像素
001Ch-001Dh	biBitCount	0008h	8bit 图像

表 3-2 BMP 信息头数据解析

1.4 调色板

1，4，8 位图像才会使用调色板数据，16,24,32 位图像不需要调色板数据，即调色板最多只需要 256 项（索引 0 - 255）。

颜色表的大小根据所使用的颜色模式而定：2 色图像为 8 字节；16 色图像位 64 字节；256 色图像为 1024 字节。其中，每 4 字节表示一种颜色，并以 B（蓝色）、G（绿色）、R（红色）、alpha（32 位位图的透明度值，一般不需要）。即首先 4 字节表示颜色号 1 的颜色，接下来表示颜色号 2 的颜色，依此类推。

颜色表中 RGBQUAD 结构数据的个数有 biBitCount 来确定，当 biBitCount=1,4,8 时，分别有 2,16,256 个表项。

当 biBitCount=1 时，为 2 色图像，BMP 位图中有 2 个数据结构 RGBQUAD，一个调色板占用 4 字节数据，所以 2 色图像的调色板长度为 2*4 为 8 字节。

当 biBitCount=4 时，为 16 色图像，BMP 位图中有 16 个数据结构 RGBQUAD，一个调色板占用 4 字节数据，所以 16 像的调色板长度为 16*4 为 64 字节。

当 biBitCount=8 时，为 256 色图像，BMP 位图中有 256 个数据结构 RGBQUAD，一个调色板占用 4 字节数据，所以 256 色图像的调色板长度为 256*4 为 1024 字节。

当 biBitCount=16，24 或 32 时，没有颜色表。

1.5 图片点阵数据

颜色表接下来位为位图文件的图像数据区，在此部分记录着每点像素对应的颜色号，其记录方式也随颜色模式而定，既 2 色图像每点占 1 位（8 位为 1 字节）；16 色图像每点占 4 位（半字节）；256 色图像每点占 8 位（1 字节）；真彩色图像每点占 24 位（3 字节）。所以，整个数据区的大小也会随之变化。究其规律而言，可的出如下计算公式：图像数据信息大小=（图像宽度*图像高度*记录像素的位数）/8。[2]

图 4 点阵数据

```
00000430  FE 00 FF FF FF 00 67 63  64 54 56 62 62 00 62 65  ......gcdTVbb.be
00000440  66 56 45 47 5F 00 61 5C  5B 63 48 47 52 00 58 4B  fVEG_.a\[cHGR.XK
00000450  55 65 5A 5B 46 00 68 47  3F 69 5D 4C 2A 00 61 59  UeZ[F.hG?i]L*.aY
00000460  5A 5F 47 28 45 00 52 52  49 3B 37 50 5A 00        Z_G(E.RRI;7PZ.
```

82	82	73	59	55	80	90
97	89	90	95	71	40	69
104	71	63	105	93	76	42
88	75	85	101	90	91	70
97	92	91	99	72	71	82
98	101	102	86	69	71	95
103	99	100	84	86	98	98

表 4 灰度值表

二. Lena 512*512 图像灰度级逐级递减 8-1 显示

2.1 问题分析，预备知识

灰度是指黑白图像中像素的颜色深度，灰度的最小值为 0 指示黑色，而最大值指示白色，不同灰度级的图像其灰度值有不同的范围，通常 8 比特图像的灰度值范围是 0-255，一幅 k 比特的图像，其灰度级为

$$L = 2^k$$

通常一幅图像的灰度级越高，显示出来的细节越清晰，但是占用的存储 Jon 关键也就越大。当一幅图像以两个灰度级显示时，实际上就成了一副只有纯黑纯白两色的 0-1 图像，此时图像中的很多信息都会损失掉。

2.2 工具

Matlab 中有专门的函数可以实现图像的基本处理，如：

（1） Imread 函数专门用于图片的读入，并将灰度信息生成一个矩阵

（2） Imshow 函数用于图片的显示，并且可以自行调整黑白对应的灰度级数值

2.3 实验结果

图 5 灰度级从 8-1 的图片显示

2.4 结果分析

有上面的 8 张图片所显示的信息中可以看出，随着灰度级的降低，图像的最大灰度值与最小灰度值之差越来越小，也就是图像的动态范围逐渐减小，图像中的细节越来越不显著，损失的信息越来越多，直观的视觉效果越来越差。

三．计算 Lena 图像的均值差

3.1 均值和方差的定义

数字图像一般以二维矩阵的形式来表示，把这些像素的灰度数据看做样本值，那么就可以引入一些统计特征来描述他们，最常用也最简单的就是均值和方差。如果以 f(x,y)表示一幅 M*N 图像的像素灰度值，那么它的均值就可以表示为

$$f=\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}f（i，j）$$

均值描述了一组数据的平均水平，用图像上也就是图像的平均灰度，均值越大，那么图像的总体色调越亮，反之越暗。若要反应灰度值的离散程度，就要用方差

$$S=\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}(f（i，j）-f)^2$$

S 的值越大，图像的对比度越大。

3.2 计算结果

```
av =

   99.0512
var =

   2.7960e+03
```

四．Lena 图像的三种内插方式

内插是在诸如放大，收缩，旋转和几何校正等许多操作中广泛使用的基本工具，它是一种利用已知数据估计未知位置数据的处理方法。例如在图像的放大处理中，原图像放大 2 倍则会有一半的位置像素值无法直接得到，这时候就需要进行内插处理来填补空缺位置。最常用的图像内插方法有最近邻内插，双线性内插和双三次线型内插。

4.1 最近邻内插[3]

这是最简单的一种插值方法，不需要计算，在待求象素的四邻象素中，将距离待求象素最近的邻象素灰度赋给待求象素。设 i+u，j+v(i，j 为正整数， u，v 为大于零小于 1 的小数，下同)为待求象素坐标，则待求象素灰度的值 f(i+u，j+v) 如下图所示：

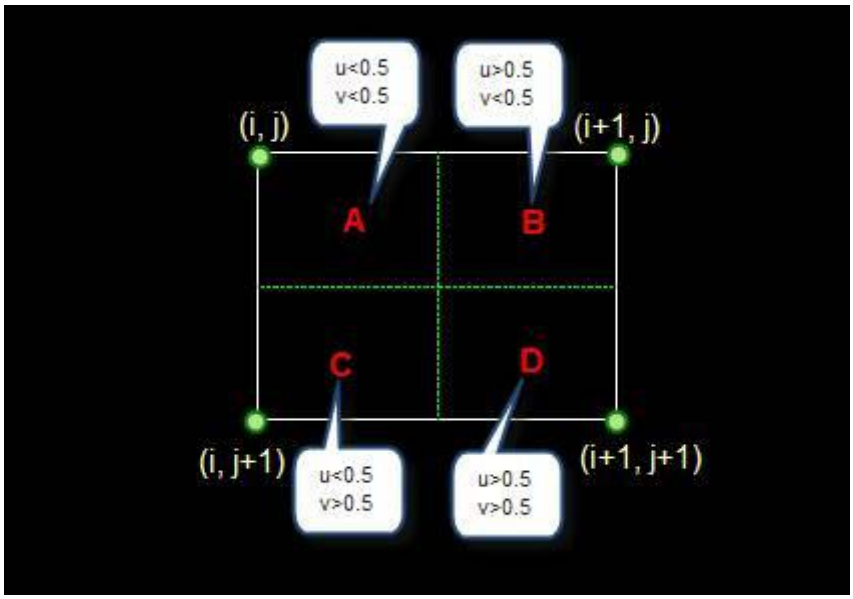


图 6

如果(i+u, j+v)落在 A 区，即 u<0.5, v<0.5，则将左上角象素的灰度值赋给待求象素，同理，落在 B 区则赋予右上角的象素灰度值，落在 C 区则赋予左下角象素的灰度值，落在 D 区则赋予右下角象素的灰度值。最邻近元法计算量较小，但可能会造成插值生成的图像灰度上的不连续，在灰度变化的地方可能出现明显的锯齿状。

4.2 双线性内插法

双线性内插法是利用待求象素四个邻象素的灰度在两个方向上作线性内插，如下图所示：

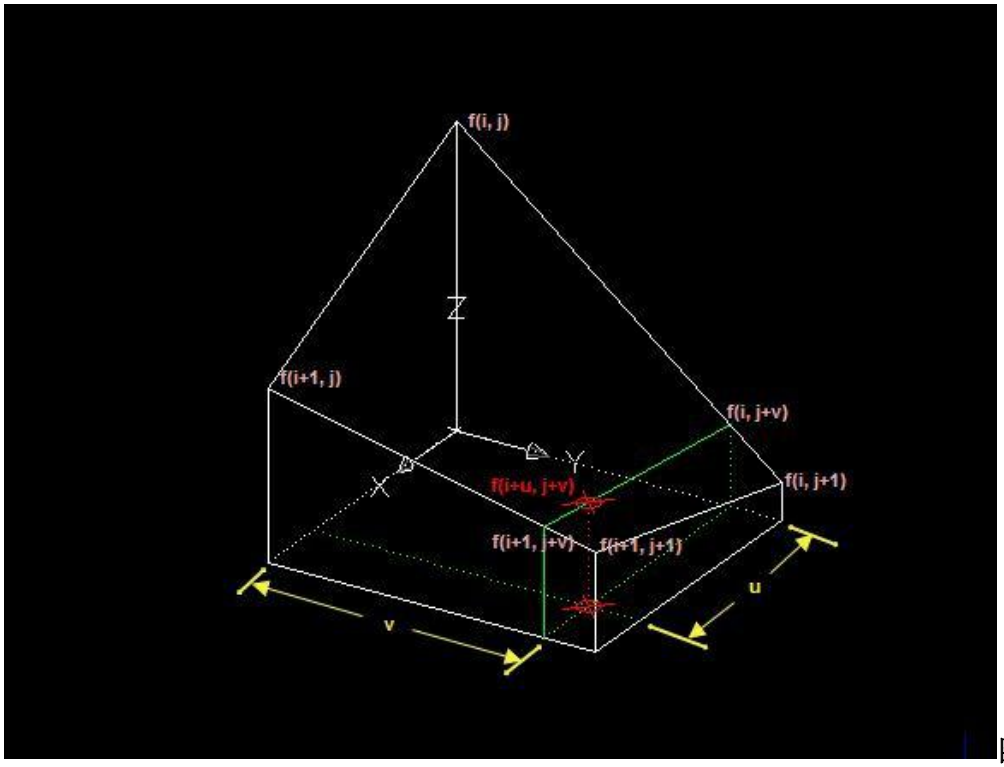


图 7

对于 (i, j+v)，f(i, j) 到 f(i, j+1) 的灰度变化为线性关系，则有：

f(i, j+v) = [f(i, j+1) - f(i, j)] * v + f(i, j)

同理对于 (i+1, j+v) 则有：

f(i+1, j+v) = [f(i+1, j+1) - f(i+1, j)] * v + f(i+1, j)

从 f(i, j+v) 到 f(i+1, j+v) 的灰度变化也为线性关系，由此可推导出待求象素灰度的计算式如下：

f(i+u, j+v) = (1-u) * (1-v) * f(i, j) + (1-u) * v * f(i, j+1) + u * (1-v) * f(i+1, j) + u * v * f(i+1, j+1)

双线性内插法的计算比最邻近点法复杂，计算量较大，但没有灰度不连续的缺点，结果基本令人满意。它具有低通滤波性质，使高频分量受损，图像轮廓可能会有一点模糊。

4.3 三次内插法

该方法利用三次多项式 S(x)求逼近理论上最佳插值函数 sin(x)/x，其数学表达式为：

待求像素(x, y)的灰度值由其周围 16 个灰度值加权内插得到，如下图：

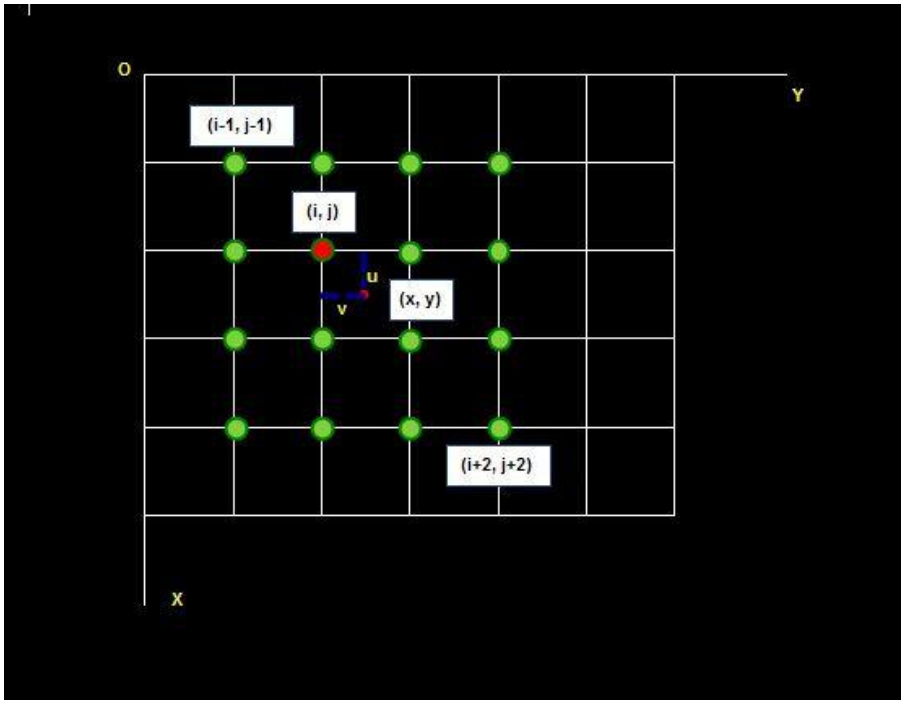


图 8

待求像素的灰度计算式如下：

f(x, y) = f(i+u, j+v) = ABC

其中:

三次曲线插值方法计算量较大，但插值后的图像效果最好。

最邻近插值（近邻取样法）、双线性内插值、三次卷积法 等插值算法对于旋转变换、错切变换、一般线性变换 和 非线性变换 都适用。

4.4 实验结果

图 9 三种内插方式的处理结果

整幅图片进行显示可能看不出差别，现在进行局部放大显示。

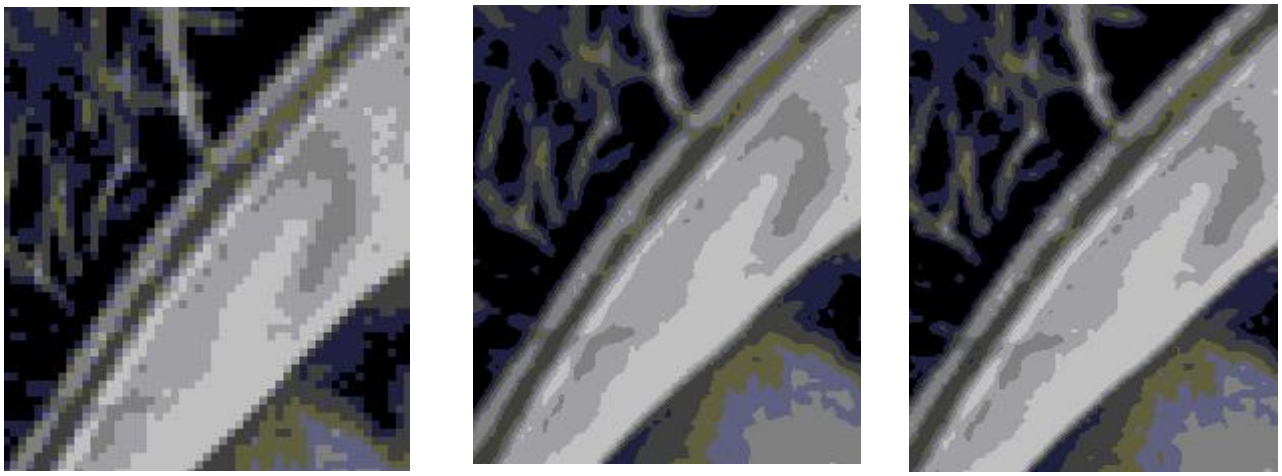


图 10 从左到右分别为最近邻，双线性以及双三次线性内插法所得图片的局部放大信息

可以看出原图像经过最近邻内插所得到的图像的曲线上呈现锯齿状，而采用双线性内插和双三次线性内插所的图像的曲线均非常平滑。

五．Lena 和 Elaine 图像的 shear，旋转和内插处理

5.1 shear 变换

剪切变换(shear transformation)是空间线性变换之一。变换后的新坐标值(x^*, y^*)如图 1 所示，相当于原坐标值(x, y)经横向剪切。其中c值为剪切常数。剪切变换可以仅是x坐标、或仅是y坐标受横向剪切，也可以是两个坐标同时受横向剪切。x坐标受横向剪切时的变换关系如图 1(b)所示，取决于下式所示的矩阵乘法运算 ：$[x^*, y^*] = [x, y]\begin{pmatrix} 1 & 0 \\ c & 1 \end{pmatrix} = [(x + cy)\, y].$ 仅x坐标或y坐标受剪切以及x与y坐标同时受剪切时的变换矩阵分别为：$\begin{pmatrix} 1 & 0 \\ c & 1 \end{pmatrix}, \begin{pmatrix} 1 & b \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & b \\ c & 1 \end{pmatrix}$.[4]

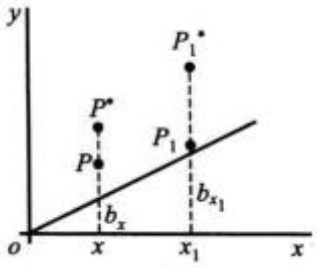


图 1(a) 剪切变换 y 向

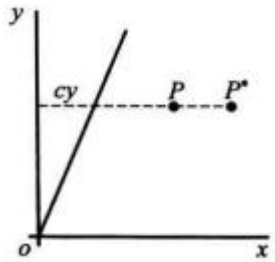


图 1(b) 剪切变换

5.2 旋转变换

一般图像的旋转是以图像的中心为原点，旋转一定的角度，也就是将图像上的所有像素都旋转一个相同的角度。旋转后图像的的大小一般会改变，即可以把转出显示区域的图像截去，或者扩大图像范围来显示所有的图像。图像的旋转变换也可以用矩阵变换来表示。设点$P_0(x_0, y_0)$逆时针旋转θ角后的对应点为$P(x, y)$。那么，旋转前后点$P_0(x_0, y_0)$、$P(x, y)$的坐标分别是：

$$\begin{cases} x_0 = r\cos\alpha \\ y_0 = r\cos\alpha \end{cases}$$

$$\begin{cases} x = r\cos(\alpha+\theta) = r\cos\alpha\cos\theta - r\sin\alpha\sin\theta = x_0\cos\theta - y_0\sin\theta \\ y = r\sin(\alpha+\theta) = r\sin\alpha\cos\theta + r\cos\alpha\sin\theta = x_0\sin\theta + y_0\cos\theta \end{cases}$$

写成矩阵表达式为

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix}$$

其逆运算为

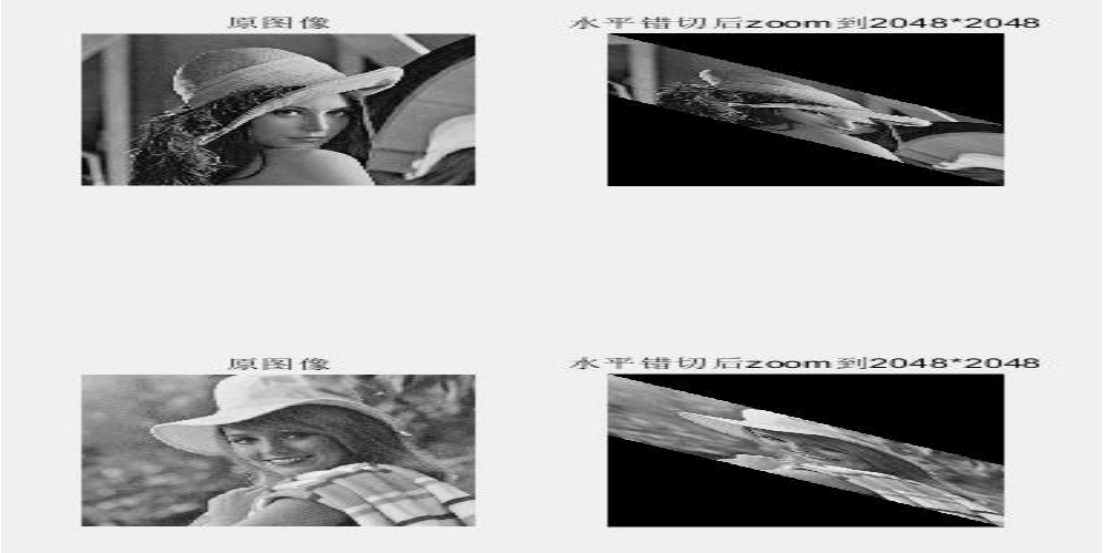
$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix}$$

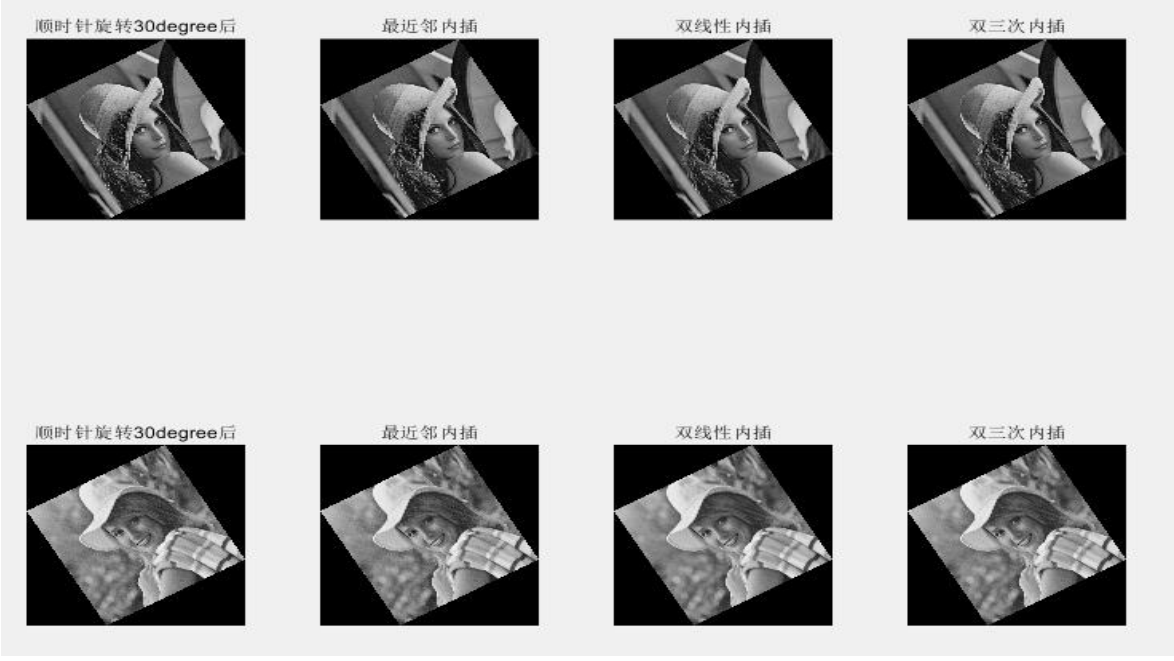
利用上述方法进行图像旋转时需要注意如下两点：

（1）图像旋转之前，为了避免信息的丢失，一定要有坐标平移。

（2）图像旋转之后，会出现许多空洞点。对这些空洞点必须进行填充处理，否则画面效果不好，一般也称这种操作为插值处理。

5.3 结果





参考文献：

[1] CSDN blog,BMP 格式， https://blog.csdn.net/zhongranxu/article/details/80259751，2018 年 5 月 9 日。

[2] CSDN blog,BMP 图像数据格式详解，

https://blog.csdn.net/testcs_dn/article/details/76719347,2017 年 8 月 5 日

[3] CSDN blog Matlab 插值算法（最邻近、双线性、双三次插值）

https://blog.csdn.net/u013146742/article/details/52923864 2016 年 10 月 25 日

[4] 百度百科 剪切变换

https://baike.baidu.com/item/%E5%89%AA%E5%88%87%E5%8F%98%E6%8D%A2/4969513?fr=aladdin