

数字图像处理

第一次作业

姓名：何佳

班级：自动化 64

学号：2160700002

提交日期：2019.3.5

摘要：本次报告首先以 7.bmp 为例简单介绍了 Bmp 图像格式及其数据结构，随后以 Matlab R2018a 为平台，完成了以下 4 个任务：①把 lena.bmp 图像作 8 到 1 的灰度级逐级递减显示；②计算 lena.bmp 图像的均值和方差；③用近邻、双线性和双三次插值法将 512*512 大小的 lena.bmp 放大到 2048*2048；④把 lena.bmp 和 elain.bmp 分别进行水平偏移变换和旋转 30 度变换，并采用近邻、双线性和双三次插值法将其放大到 2048*2048。以上的任务均得到良好的结果。

1. Bmp 图像格式简介，以 7.bmp 为例说明

1.1 BMP 文件格式简介

BMP（全称 Bitmap）是 Windows 操作系统中的标准图像文件格式，可以分成两类：设备有向量相关位图（DDB）和设备无向量相关位图（DIB），使用非常广。它采用位映射存储格式，除了图像深度可选以外，不采用其他任何压缩，因此，BMP 文件所占用的空间很大。BMP 文件的图像深度可选 1bit、4bit、8bit 及 24bit。BMP 文件存储数据时，图像的扫描方式是按从左到右、从下到上的顺序。由于 BMP 文件格式是 Windows 环境中交换与图有关的数据的一种标准，因此在 Windows 环境中运行的图形图像软件都支持 BMP 图像格式。

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②位图信息头③调色板④位图数据，下面分别进行介绍。

1.1.1 bmp 文件头(bitmap file header)

bmp 文件头提供文件的格式、大小等信息，共 14 个字节。Windows 为 bmp 文件头定义了如下结构体：

```
typedef struct tagBITMAPFILEHEADER
{
    UINT16 bfType;
    DWORD bfSize;
    UINT16 bfReserved1;
    UINT16 bfReserved2;
    DWORD bfOffBits;
} BITMAPFILEHEADER;
```

其中：

变量名	地址偏移	大小	作用
bfType	0000h	2 bytes	说明文件的类型，可取值为： • 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
bfSize	0002h	4 bytes	说明该位图文件的大小，用字节为单位
bfReserved1	0006h	2 bytes	保留，必须设置为0
bfReserved2	0008h	2 bytes	保留，必须设置为0
bfOffBits	000Ah	4 bytes	说明从文件头开始到实际的图象数据之间的字节的偏移量。 这个参数是非常有用的，因为位图信息头和调色板的长度会根据不同情况而变化，所以我们可以用这个偏移值迅速的从文件中读取到位图数据。

1.1.2 位图信息头(bitmap information)

位图信息头提供图像数据的尺寸、位平面数、压缩方式、颜色索引等信息，其结构体如下：

变量名	地址偏移	大小	作用
biSize	000Eh	4 bytes	BITMAPINFOHEADER结构所需要的字数。
biWidth	0012h	4 bytes	说明图像的宽度，用像素为单位
biHeight	0016h	4 bytes	说明图像的高度，以像素为单位。 注：这个值除了用于描述图像的高度之外，它还有另一个用处，就是指明该图像是倒向的位图，还是正向的位图。 如果该值是一个正数，说明图像是倒向的，如果该值是一个负数，则说明图像是正向的。 大多数的BMP文件都是倒向的位图，也就是高度值是一个正数。
biPlanes	001Ah	2 bytes	为目标设备说明颜色平面数， 其值将总是被设为1。
biBitCount	001Ch	2 bytes	说明比特数/像素，其值为1、4、8、16、24或32。
biCompression	001Eh	4 bytes	说明图像数据压缩的类型。取值范围： 0 BI_RGB 不压缩（最常用） 1 BI_RLE8 8比特游程编码（RLE），只用于8位位图 2 BI_RLE4 4比特游程编码（RLE），只用于4位位图 3 BI_BITFIELDS 比特域，用于16/32位位图 4 BI_JPEG JPEG 位图含JPEG图像（仅用于打印机） 5 BI_PNG PNG 位图含PNG图像（仅用于打印机）
biSizeImage	0022h	4 bytes	说明图像的大小， 以字节为单位。当用BI_RGB格式时，可设置为0。
biXPelsPerMeter	0026h	4 bytes	说明水平分辨率，用像素/米表示，有符号整数
biYPelsPerMeter	002Ah	4 bytes	说明垂直分辨率，用像素/米表示，有符号整数
biClrUsed	002Eh	4 bytes	说明位图实际使用的彩色表中的颜色索引数 （设为0的话，则说明使用所有调色板项）
biClrImportant	0032h	4 bytes	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。

1.1.3 调色板(color palette)

调色板为可选项，其实质是一张映射表，标识颜色索引号与其代表的颜色的对应关系。它在文件中的布局就像一个二维数组 palette[N][4]，其中 N 表示总的颜色索引数，每行的四个元素分别表示该索引对应的 B、G、R 和 Alpha 的值，每个分量占一个字节。如不设透明通道时，Alpha 为 0。

调色板数据可这样表示：索引：(蓝，绿，红，Alpha)，以作为解读。

1.1.4 位图数据(bitmap data)

位图数据就是图像数据，每个像素占一个字节，取得这个字节后，以该字节为索引查询相应的颜色，并显示到相应的显示设备上就可以了。

1.2 以 7.bmp 为例分析 bmp 文件格式

1.2.1 采用 16 进制文本编辑器得到位图信息

本次作业采用“Free Hex Editor Neo 6.31”打开 bmp 文件，得到以下信息：

00000000	42	4d	6e	04	00	00	00	00	00	36	04	00	00	28	00
00000010	00	00	07	00	00	00	07	00	00	00	01	00	08	00	00
00000020	00	00	38	00	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	00	00	00	00	00	00	00	00	01	01	01	00	02
00000040	02	00	03	03	03	00	04	04	04	00	05	05	05	00	06
00000050	06	00	07	07	07	00	08	08	08	00	09	09	09	00	0a
00000060	0a	00	0b	0b	0b	00	0c	0c	0c	00	0d	0d	0d	00	0e
00000070	0e	00	0f	0f	0f	00	10	10	10	00	11	11	11	00	12
00000080	12	00	13	13	13	00	14	14	14	00	15	15	15	00	16
00000090	16	00	17	17	17	00	18	18	18	00	19	19	19	00	1a
000000a0	1a	00	1b	1b	1b	00	1c	1c	1c	00	1d	1d	1d	00	1e
000000b0	1e	00	1f	1f	1f	00	20	20	20	00	21	21	21	00	22
000000c0	22	00	23	23	23	00	24	24	24	00	25	25	25	00	26
000000d0	26	00	27	27	27	00	28	28	28	00	29	29	29	00	2a
000000e0	2a	00	2b	2b	2b	00	2c	2c	2c	00	2d	2d	2d	00	2e
000000f0	2e	00	2f	2f	2f	00	30	30	30	00	31	31	31	00	32
00000100	32	00	33	33	33	00	34	34	34	00	35	35	35	00	36
00000110	36	00	37	37	37	00	38	38	38	00	39	39	39	00	3a
00000120	3a	00	3b	3b	3b	00	3c	3c	3c	00	3d	3d	3d	00	3e
00000130	3e	00	3f	3f	3f	00	40	40	40	00	41	41	41	00	42
00000140	42	00	43	43	43	00	44	44	44	00	45	45	45	00	46
00000150	46	00	47	47	47	00	48	48	48	00	49	49	49	00	4a
00000160	4a	00	4b	4b	4b	00	4c	4c	4c	00	4d	4d	4d	00	4e
00000170	4e	00	4f	4f	4f	00	50	50	50	00	51	51	51	00	52
00000180	52	00	53	53	53	00	54	54	54	00	55	55	55	00	56
00000190	56	00	57	57	57	00	58	58	58	00	59	59	59	00	5a
000001a0	5a	00	5b	5b	5b	00	5c	5c	5c	00	5d	5d	5d	00	5e
000001b0	5e	00	5f	5f	5f	00	60	60	60	00	61	61	61	00	62
000001c0	62	00	63	63	63	00	64	64	64	00	65	65	65	00	66
000001d0	66	00	67	67	67	00	68	68	68	00	69	69	69	00	6a
000001e0	6a	00	6b	6b	6b	00	6c	6c	6c	00	6d	6d	6d	00	6e
000001f0	6e	00	6f	6f	6f	00	70	70	70	00	71	71	71	00	72
00000200	72	00	73	73	73	00	74	74	74	00	75	75	75	00	76
00000210	76	00	77	77	77	00	78	78	78	00	79	79	79	00	7a
00000220	7a	00	7b	7b	7b	00	7c	7c	7c	00	7d	7d	7d	00	7e
00000230	7e	00	7f	7f	7f	00	80	80	80	00	81	81	81	00	82
00000240	82	00	83	83	83	00	84	84	84	00	85	85	85	00	86
00000250	86	00	87	87	87	00	88	88	88	00	89	89	89	00	8a
00000260	8a	00	8b	8b	8b	00	8c	8c	8c	00	8d	8d	8d	00	8e
00000270	8e	00	8f	8f	8f	00	90	90	90	00	91	91	91	00	92
00000280	92	00	93	93	93	00	94	94	94	00	95	95	95	00	96
00000290	96	00	97	97	97	00	98	98	98	00	99	99	99	00	9a
000002a0	9a	00	9b	9b	9b	00	9c	9c	9c	00	9d	9d	9d	00	9e
000002b0	9e	00	9f	9f	9f	00	a0	a0	a0	00	a1	a1	a1	00	a2
000002c0	a2	00	a3	a3	a3	00	a4	a4	a4	00	a5	a5	a5	00	a6
000002d0	a6	00	a7	a7	a7	00	a8	a8	a8	00	a9	a9	a9	00	aa
000002e0	aa	00	ab	ab	ab	00	ac	ac	ac	00	ad	ad	ad	00	ae
000002f0	ae	00	af	af	af	00	b0	b0	b0	00	b1	b1	b1	00	b2
00000300	b2	00	b3	b3	b3	00	b4	b4	b4	00	b5	b5	b5	00	b6
00000310	b6	00	b7	b7	b7	00	b8	b8	b8	00	b9	b9	b9	00	ba
00000320	ba	00	bb	bb	bb	00	bc	bc	bc	00	bd	bd	bd	00	be
00000330	be	00	bf	bf	bf	00	c0	c0	c0	00	c1	c1	c1	00	c2
00000340	c2	00	c3	c3	c3	00	c4	c4	c4	00	c5	c5	c5	00	c6
00000350	c6	00	c7	c7	c7	00	c8	c8	c8	00	c9	c9	c9	00	ca
00000360	ca	00	cb	cb	cb	00	cc	cc	cc	00	cd	cd	cd	00	ce
00000370	ce	00	cf	cf	cf	00	d0	d0	d0	00	d1	d1	d1	00	d2
00000380	d2	00	d3	d3	d3	00	d4	d4	d4	00	d5	d5	d5	00	d6
00000390	d6	00	d7	d7	d7	00	d8	d8	d8	00	d9	d9	d9	00	da
000003a0	da	00	db	db	db	00	dc	dc	dc	00	dd	dd	dd	00	de
000003b0	de	00	df	df	df	00	e0	e0	e0	00	e1	e1	e1	00	e2
000003c0	e2	00	e3	e3	e3	00	e4	e4	e4	00	e5	e5	e5	00	e6
000003d0	e6	00	e7	e7	e7	00	e8	e8	e8	00	e9	e9	e9	00	ea
000003e0	ea	00	eb	eb	eb	00	ec	ec	ec	00	ed	ed	ed	00	ee
000003f0	ee	00	ef	ef	ef	00	f0	f0	f0	00	f1	f1	f1	00	f2
00000400	f2	00	f3	f3	f3	00	f4	f4	f4	00	f5	f5	f5	00	f6
00000410	f6	00	f7	f7	f7	00	f8	f8	f8	00	f9	f9	f9	00	fa
00000420	fa	00	fb	fb	fb	00	fc	fc	fc	00	fd	fd	fd	00	fe
00000430	fe	00	ff	ff	ff	00	67	63	64	54	56	62	62	00	65
00000440	66	56	45	47	5f	00	61	5c	5b	63	48	47	52	00	4b
00000450	55	65	5a	5b	46	00	68	47	3f	69	5d	4c	2a	00	61
00000460	5a	5f	47	28	45	00	52	52	49	3b	37	50	5a	00	..

然后结合之前的 bmp 文件数据结构，对特定位置的 16 进制编码进行分析。

1.2.2 bmp 文件头

1-2: 424dh="BM", 表示这是 Windows 支持的位图格式
3-5: 0000046eh= 1134 B=1.134KB, 表明文件大小为 1.134KB
6-9: 00 00 00 00, 两个保留段, 符合位图格式规范
A-D: 00000436h=1078, 即从文件头到位图数据需偏移 1078 字节

1.2.3 位图信息头

0E-11: 00000028h = 40, 说明该文件位图信息头的大小为 40 个字节。
12-15: 00000007h = 7, 图像宽为 7 像素, 与文件属性一致。
16-19: 00000007h = 7, 图像高为 7 像素, 与文件属性一致。这是一个正数, 说明图像数据是从图像左下角到右上角排列的。
1A-1B: 0001h, 该值总为 1。
1C-1D: 0008h = 8, 表示每个像素占 8 个比特, 即该图像共有 256 种颜色。
1E-21: 00000000h, BI_RGB, 说明本图像不压缩。
22-25: 00000038h=56B, 图像的大小为 56B 。
26-29: 00000000h, 水平分辨率, 缺省。
2A-2D: 00000000h, 垂直分辨率, 缺省。
2E-31: 00000000h = 0, 说明使用所有调色板项
32-35: 00000000h = 0, 说明本位图的颜色索引数都重要。

1.2.4 调色板

0 号: (00, 00, 00, 00)
1 号: (01, 01, 01, 00)
2 号: (02, 02, 02, 00)
.....
255 号: (ff, ff, ff, 00)

一共有 256 种颜色, 每个颜色占用 4 个字节, 就是一共 1024 个字节, 再加上前面的文件信息头和位图信息头的 54 个字节加起来一共是 1078 个字节。也就是说在位图数据出现之前一共有 1078 个字节, 与我们在文件信息头得到的信息: 文件头到文图数据区的偏移为 1078 个字节一致。

1.2.5 位图数据

剩余的 56 个字节即为位图数据, 取得这个字节后, 以该字节为索引查询相应的颜色, 并显示到相应的显示设备上就可以了。而 7.bmp 文件为 $7 \times 7 = 49$ 像素, 实际位图数据却是 56 个字节, 观察数据发现每隔 7 个字节就会出现一个 00, 因此猜想位图存储和读取是以 8 个字节为单位, 这样一次读取 8 个字节时, 给 7 像素后面补一个 00 可以使每次读取均恰好代表了一行像素或一列像素, 使得原先的 $7 \times 7 = 49$ 字节变成 $7 \times 8 = 56$ 字节。

2. 把 lena.bmp 从 8 到 1 进行灰度级逐级递减显示

2.1 问题分析

在不改变图像大小的前提下, 若采用逐级递减的方式, 从 8bit (256) 灰度级到 1bit (2) 灰度级需要连续 7 次对逐像素的灰度值进行除以 2 的操作, 因此采用循环次数为 7 的循环体实现这一操作。当灰度值进行了若干次除 2 操作后, 在 Matlab 中 imshow()显示图像时需要将显示的灰度范围做以调整, 即采用 imshow(Image,[a,b])的函数, a 和 b 分别为灰度显示的上下界。

本问题 matlab 实现代码参见后面“源代码”部分内容。

2.2 结果分析与讨论

运行程序后得到的结果图如下：

lena-8bit



lena-7bit



lena-6bit



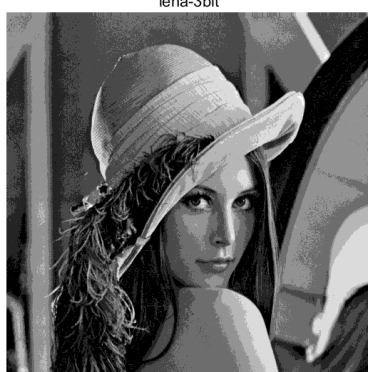
lena-5bit

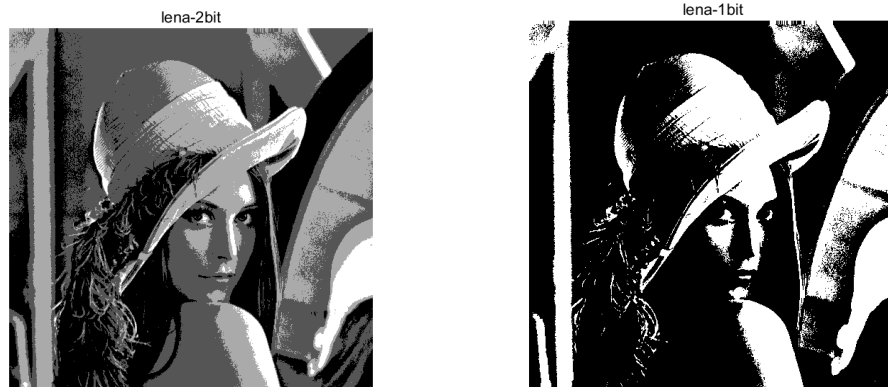


lena-4bit



lena-3bit





从上图对比可知，256 级、128 级、64 级以及 32 级灰度的图像几乎没有太大的区别；然而在灰度级为 16 的图“lena-4bit”中出现了较为明显的伪轮廓，这种效果是由数字图像的平滑区域中的灰度级数不足引起的。

3. 计算 lena.bmp 图像的均值和方差

3.1 问题分析

在 matlab 中读入 Bmp 图像后会存储为以像素长宽为大小的、以灰度值为元素的矩阵，求 bmp 图像的均值即为求整个矩阵的均值，同理求 bmp 图像的方差即为求整个矩阵所有元素的方差。

本问题 matlab 实现代码参见后面“源代码”部分内容。

3.2 结果分析与讨论

运行程序后得到的均值 $\text{mean} = 99.0512$ ；方差 $\text{d2} = 52.8775$ ，均值说明了 lena.bmp 的图像的平均灰度值为 99.0512，方差衡量了 lena.bmp 的对比度大小。

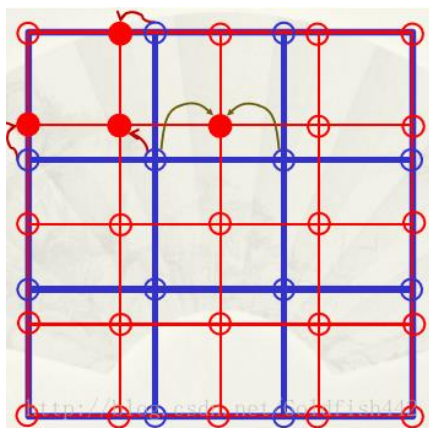
4. 把 lena.bmp 图像用最近邻、双线性插值和双三次插值法放大到 2048*2048

4.1 问题分析

lena.bmp 图像原大小为 512*512，因此长宽分别放大 4 倍即可。

4.1.1 最近邻插值法

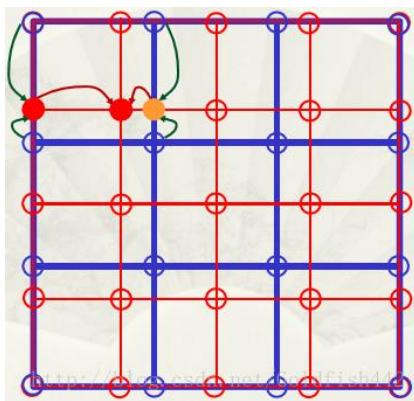
最近邻插值法是利用距离最近的点去代替要变换的点，如下图，蓝色框上每个点代表原图对应位置的灰度值，红色为放大后的点。



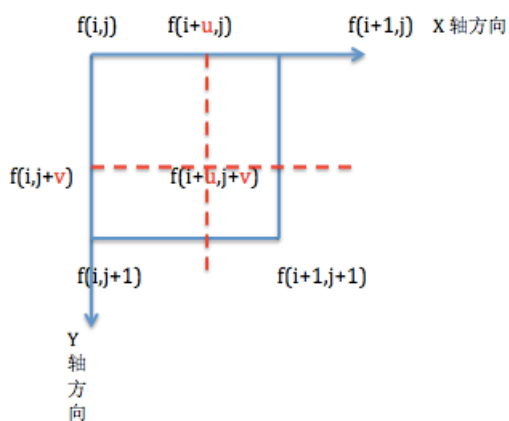
在 matlab 中可直接调用 `imresize(Image,n,'nearest')` 函数进行放大, n 表示放大倍数, 也可根据原理直接编写。本问题 matlab 实现代码参见后面“源代码”部分内容。

4.1.2 双线性插值法

双线性内插法是利用最近的四个点, 进行线性映射, 如下图:



利用矩阵中的四个像素值计算新的像素值的计算方法如下:

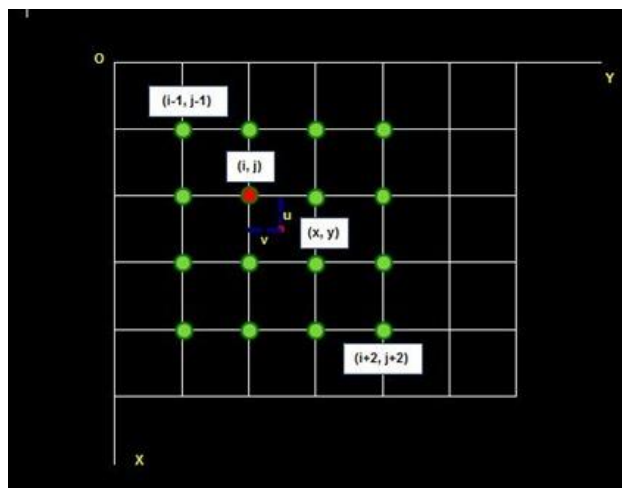


$$\text{则 } f(i+u, j+v) = (1-u)(1-v)f(i, j) + u(1-v)f(i+1, j) + (1-u)v f(i, j+1) + uv f(i+1, j+1)$$

在 matlab 中可直接调用 `imresize(Image,n,'bilinear')` 函数进行放大, n 表示放大倍数, 也可根据原理直接编写。本问题 matlab 实现代码参见后面“源代码”部分内容。

4.1.3 双三次插值法

双三次内插法是待求像素 (x, y) 的灰度值由其周围 16 个灰度值加权内插得到, 如下图:



在 matlab 中可直接调用 `imresize(Image,n,'bicubic')` 函数进行放大, n 表示放大倍数, 也可根据原理直接编写。本问题 matlab 实现代码参见后面“源代码”部分内容。

4.2 结果分析与讨论

运行程序后得到的结果图如下 (为方便对比, 以下将放大后的图像调整为与原图相同大小):



对比上面 4 幅图可知: 经过最近邻内插把原图像中最近邻的灰度赋给了每个新位置, 所得的图像的肩膀以及脸顿部分的曲线呈现锯齿状, 产生了不希望的人为缺陷; 双线性内插法

用 4 个最近邻去估计给定位置的灰度，所得的图像的肩膀以及脸颊部分的曲线则比较平滑，基本没有锯齿出现，但计算量也相对有所增加；而双三次内插法用 16 个最近邻点的灰度去估计给定位置的灰度，所得的图像的曲线非常平滑，在保持细节方面最佳，但计算量也最大。

5.把 lena.bmp 和 elain1.bmp 图像分别进行水平偏移和旋转 30 度，并采用近邻、双线性和双三次插值法放大到 2048*2048

5.1 问题分析

数字图像的二维仿射变换一般包括了恒等变换、尺度变换、旋转变换、平移变换、垂直偏移变换以及水平偏移变换。本题目涉及水平偏移变换和旋转变换。

在 matlab 中实现数字图像的二维空间变换包括以下步骤：

- ①定义空间变换的参数
- ②创建变换结构体，它包含了执行变换需要的所有参数。可定义的空间变换包括仿射变换 affine transformations (如平移 translation，缩放 scaling，旋转 rotation，剪切 shearing)、投影变换 projective transformations 和自定义的变换 custom transformations。创建结构体的方法有两种：使用 maketform 或者使用 cp2tform。

以仿射变换为例，可通过 TFORM=maketform('affine',image);得到变换结构体

- ③执行变换
- 通过将要变换的图像和 TFORM 结构体传递给 imtransform 函数即可实现变换。即：
NewImage=imtransform(Image,TFORM);

采用近邻、双线性和双三次插值法放大到 2048*2048 的解决办法参见上个问题。

5.1.1 水平偏移变换

偏移变换所需仿射矩阵如下：


Shear		$\begin{bmatrix} 1 & sh_y & 0 \\ sh_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	<p>sh_x specifies the shear factor along the x axis</p> <p>sh_y specifies the shear factor along the y axis.</p>
-------	---	---	--

对于水平偏移变换，shy=0，本次作业 shx 参数选用 1。坐标变换为：x=v ; y=shx*v+w=v+w，其中(v,w)为输入图像的像素，(x,y)为输出图像的相应像素。

本问题 matlab 实现代码参见后面“源代码”部分内容。

5.1.2 旋转变换

旋转变换所需仿射矩阵如下：

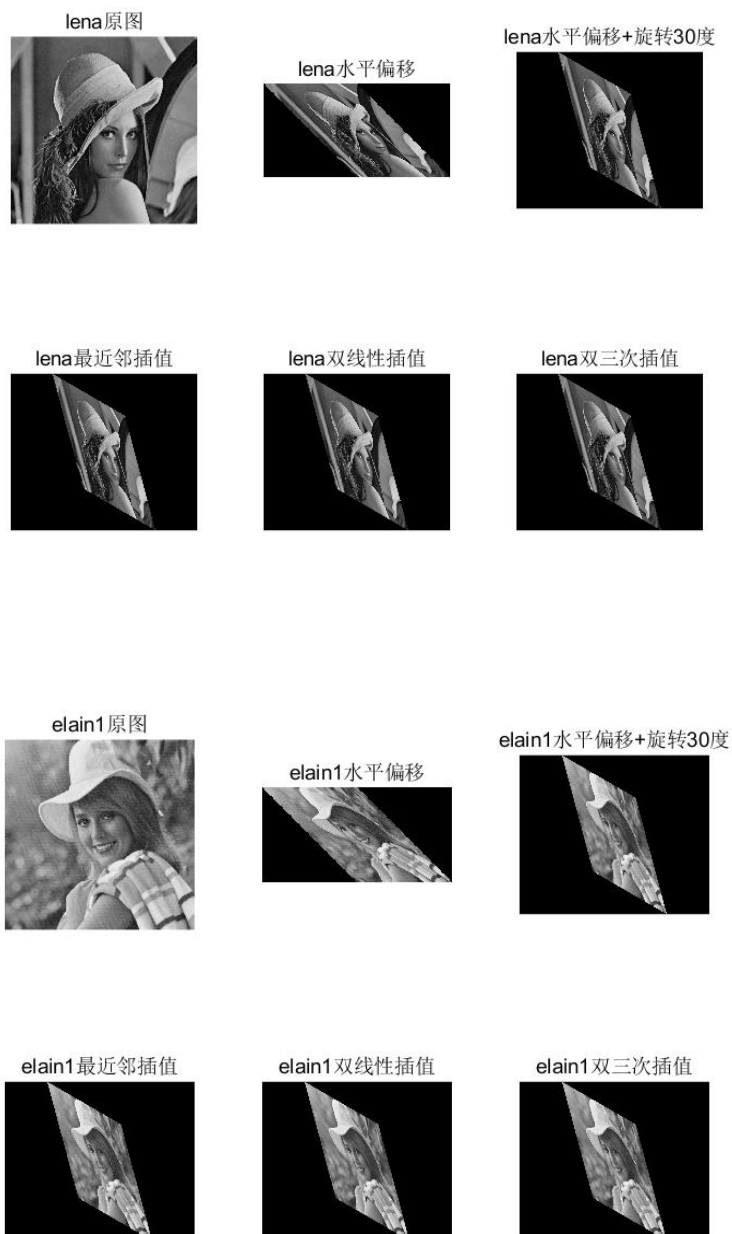
Rotation		$\begin{bmatrix} \cos(q) & \sin(q) & 0 \\ -\sin(q) & \cos(q) & 0 \\ 0 & 0 & 1 \end{bmatrix}$	<p>q specifies the angle of rotation.</p>
----------	---	--	---

本次作业要求为旋转 30 度，即上述 q=30.

本问题 matlab 实现代码参见后面“源代码”部分内容。

5.2 结果分析与讨论

运行程序后得到的结果图如下（为方便对比，将放大后的图像调整为与原图相同大小进行显示）：



附录

参考文献

<https://baike.baidu.com/item/BMP/35116?fr=aladdin>
<https://www.cnblogs.com/wainiwann/p/7086844.html>
https://blog.csdn.net/Justin_chuanyuan/article/details/71078161
<https://blog.csdn.net/Goldfish442/article/details/61933735>

<https://www.cnblogs.com/bigpo/p/4126806.html>

<https://blog.csdn.net/hd19890207/article/details/73610545>

源代码

1.无代码

2. 《Grayscale_01.m》

```
LENA=imread('lena.bmp');
figure(1)
imshow(LENA);
title('lena-8bit');
[x,y]=size(LENA);
for i=2:8
    for j=1:x
        for k=1:y
            LENA(j,k)=floor(LENA(j,k)/2);
        end
    end
    figure(i)
    imshow(LENA,[1,2.^(9-i)]);
    title(['lena-' num2str(9-i) 'bit']);
end
```

3. 《MeanVariance_02.m》

```
LENA=imread('lena.bmp');
mean=mean2(LENA)
d2=std2(LENA)
```

4. 《Interpolation_03.m》

```
LENA=imread('lena.bmp');
figure(1)
imshow(LENA);
title('lena-原图');
%最近邻插值
New1=zeros(2048,2048);
for i=1:2048
    for j=1:2048
        x=round(i/4);y=round(j/4);
        if x==0
            x=1;
        end
        if y==0
            y=1;
        end
        New1(i,j)=LENA(x,y);
    end
end
```

```

figure(2)
New1=uint8(New1);
imshow(New1);
title('lena-最近邻插值');
%双线性插值
New2=imresize(LENA,4,'bilinear');
figure(3)
imshow(New2);
title('lena-双线性插值');
%双三次插值
New3=imresize(LENA,4,'bicubic');
figure(4)
imshow(New3);
title('lena-双三次插值');
5. 《ShearRotation_04.m》
LENA=imread('lena.bmp');
ELAIN1=imread('elain1.bmp');
for i=1:2
    if i==1
        Image=LENA;
        str='lena';
    else
        Image=ELAIN1;
        str='elain1';
    end
    figure(i);
    subplot(2,3,1);
    imshow(Image);
    title([str '原图']);
    %水平偏移
    shr=[1 0 0;1 1 0;0 0 1];
    t=maketform('affine',shr);
    NewShr=imtransform(Image,t);
    subplot(2,3,2);
    imshow(NewShr);
    title([str '水平偏移']);
    %旋转30度
    rtt=[cosd(30) sind(30) 0;-sind(30) cosd(30) 0;0 0 1];
    t=maketform('affine',rtt);
    NewShrRtt=imtransform(NewShr,t);
    subplot(2,3,3);
    imshow(NewShrRtt);
    title([str '水平偏移+旋转30度']);
    %最近邻插值

```

```
New1=imresize(NewShrRtt,4,'nearest');
subplot(2,3,4);
imshow(New1);
title([str '最近邻插值']);
%双线性插值
New2=imresize(NewShrRtt,4,'bilinear');
subplot(2,3,5);
imshow(New2);
title([str '双线性插值']);
%双三次插值
New3=imresize(NewShrRtt,4,'bicubic');
subplot(2,3,6);
imshow(New3);
title([str '双三次插值']);
end
```