1、教材问题 2.10

1）高清晰度电视使用 1080 条水平电视线隔行扫描来产生图像，每两场形成一帧，每场用时 1/16 秒。图像的宽高比是 16:9。在水平行数固定的情况下，求图像的垂直分辨率。

答：图像的横向（竖向）像素数=打印横向（竖向）分辨率×打印的横向（竖向）尺寸。这里问的分辨率的单位应该是像素/条？那么，图像的横向（竖向）像素数=打印横向（竖向）分辨率×打印的横向（竖向）条数。竖向条数 1080（条）*16/9 = 1920（条）；所以垂直分辨率 = 1920（条）/竖向像素数。一般就记作分辨率 1920 了吧。

2）一家公司已经设计了一种图像获取系统，该系统由 HDTV 图像生成数字图像。在该系统中，每条（水平）电视行的分辨率与图像的宽高比成正比，彩色图像的每个像素都有 24 比特的灰度分辨率，红色、绿色、蓝色图像各 8 比特。这三幅原色图像形成彩色图像。存储 90 分钟的一部 HDTV 电影需要多少比特？

答：每场用 1/16s，则每帧用时 2×1/16=1/8 秒。

即该系统每 1/8 秒的时间形成一幅 1920×1080 分辨率的，红、绿、蓝每个像素都有 8 比特的图像。

又因为 90min 为 5400 秒,故所需的空间是

1920×1080×8×3/(1/8)×5400 = 2 149 908 480 000 bits = 2.15*10^12 bits

2、编写 MATLAB 代码，通过双线性和双三次插值来缩放和缩小图像。程序的输入是：（i）图像，（ii）沿图像行和列的缩放参数，以及（iii）插值方法。

可以直接调用函数 ZI = interp2(X,Y,Z,XI,YI,method)

用指定的算法 method 计算二维插值：

’linear’：双线性插值算法(缺省算法);

’nearest’：最临近插值;

’spline’：三次样条插值;

’cubic’：双三次插值。

但是，为了掌握算法原理，我在文件夹“2”自己实现了一遍。函数 interpolation，它在文件 interpolation.m 中给出，参数是图像文件名、高和宽的缩放系数、图像的插值方法（字符串’bilinear’或’bicubic’，不区分大小写)；无返回值，而显示缩放后的图像。

文件 bilinear.m 做双线性插值，文件 bicubic.m 做双三次插值，它们的参数是图像矩阵、高和宽的缩放系数；返回值是缩放后的图像矩阵。函数 bicubic 中还用到了插值所需要的 sw 函数，它在文件 sw.m 中定义。

3、编写 MATLAB 代码，用于计算图像的仿射变换。程序的输入是：（i）图像，（ii）仿射变换的参数，以及（iii）插值方法。

同样可以直接用函数实现。例如 3.5 可以直接调用函数：

```
I = imread('library.jpg');
T = maketform('affine',[0.3 0.1 0;0.5 0.9 0;0 0 1]);
```

```
I2 = imtransform(I,T);
T = maketform('affine',inv([0.3 0.1 0;0.5 0.9 0;0 0 1]));
I3 = imtransform(I2,T);
imshow(I)
imshow(I2)
imshow(I3)
```

得到文件夹“3”的 figure4 和 figure5。

同样为了掌握算法原理，我自己实现了一遍。函数 affirm，它在文件 affirm.m 中给出，参数是图像文件名、仿射变换矩阵、图像的插值方法（字符串’bilinear’或’bicubic’，不区分大小写）；无返回值，而显示缩放后的图像。

文件 bilinear.m 做双线性插值，文件 bicubic.m 做双三次插值，它们的参数是图像矩阵、仿射变换矩阵；返回值是缩放后的图像矩阵。函数 bicubic 中还用到了插值所需要的 sw 函数，它在文件 sw.m 中定义。

做一些变换，例如，只是进行旋转，得到的结果是正确的；但是有时候，得到的矩阵的旋转/错切方向跟正确的结果不一样。这可能是把 x 和 y 坐标搞反了导致的。

3.2 使用命令

```
a = cos(pi/4);
b = sin(pi/4);
T = [a -b 0;b a 0;0 0 1];
affirm('Lecture Hall.jpg',T,'bilinear');
```

得到 figure3.jpg

3.3 使用命令

```
T = [0.3 0.1 0;0.5 0.9 0;0 0 1];
affirm('library.jpg',T,'bilinear');
```

3.4 T = inv([0.3 0.1 0;0.5 0.9 0;0 0 1]);

```
affirm('figure4.jpg',T,'bilinear');
```

3.5 首先，尺寸变大了。这是因为第一次仿射变换时，旋转操作使得图像的面积扩大了，空余的部分用黑色填充。第二次做逆变换，旋转操作又使得图像的面积扩大了，最后可以看到，有效部分只在中间，它基本变回了原来的形状，但是周围多出了一圈填充的黑色部分。另外，因为数字图像是离散的，变换得到的图像会有空隙，这些空隙靠插值补充完整；逆变换回来时，有些原图的像素点被插值的像素点取代了，所以跟原图有差异，比如原来笔直的栏杆等现在产生了扭曲的效果。做差使用命令：

```
I = imread('library.jpg');
I2 = imread('figure5.jpg');
[h,w,d] = size(I);
[h2,w2,d2] = size(I2);
extendI = -ones(h2,w2,d2);
startH = floor((h2-h)/2);
startW = floor((w2-w)/2);
extendI(startH+1:startH+h, startW+1:startW+w, :) = I;
extendI = uint8(extendI);
delta = extendI-I2;
```

imwrite(delta, 'figure6.jpg');

得到的图中间为有效部分，其中大部分都是黑色，表示两幅图的大部分都是相同的，剩下微微能看到整幅图的轮廓，表示差异都出现在边缘处（即像素突变处）。

4、教材中的习题三（从 120 页，即 pdf 的 130 页开始）问题 3.7，3.8，3.11

3.7 对一副图像进行第二次直方图均衡处理的结果与第一次直方图均衡处理的结果相同。

设原图有 L-1 个灰度级，其频率为 a[0]~a[L-1]，所有灰度级的频率大于 0，和为 1。

| | | | | |
|---|---|---|---|---|
| 灰度级 | 0 | 1 | … | L-1 |
| 频率 | a[0] | a[1] | … | a[L-1] |

然后使用某个 T(r)函数进行变换。T(r)满足下列两个条件：

（1）T(r)在定义域中为单值且单调递增，即使得灰度级的相对大小关系不变。

（2）定义域跟值域相同，即不能改变动态范围。

T(r)一般是(L-1)乘以频率的前缀和：由于各个频率都大于 0，所以 a[i]序列的前缀和严格单增；所有频率的和为 1，所以动态范围仍是[0,L-1]。

得到的 L 个新灰度级的顺序仍是从小到大排列、两两不同的。

| | | | | |
|---|---|---|---|---|
| 新灰度级 | b[0] | b[1] | … | b[L-1] |

因为一一对应关系，新灰度级的频率依然是：

| | | | | |
|---|---|---|---|---|
| 频率 | a[0] | a[1] | … | a[L-1] |

那么如果再做一次直方图均衡化，同理，得到的新灰度级频率仍然是：

| | | | | |
|---|---|---|---|---|
| 频率 | a[0] | a[1] | … | a[L-1] |

所以，第二次处理的结果跟第一次相同。

3.8 还是考虑采用变上限积分作为 T(r)，但是下限不是 0 而是-∞。它在定义域上为单值且单调递增。

对于正态分布 r~N(m, σ^2)，可知(r-m)/σ~N(0,1)，它的无穷积分结果为 1。那么，r 的无穷积分结果为 σ。这将(-∞, ∞)对应到了[0, σ]，不符合要求。

考虑现实一些的、定义域是有穷区间的情况。假设定义域是[m-M,m+M]，M 足够大；对应的值域仍然可以看作是[0, σ]。那么，在做了变上限积分以后，还需要做一个线性变换。

综上，我选择的$T(r)=\frac{2M}{\sigma}\int_{-\infty}^{r}p_r(x)dx+m-M$（M 是一个足够大的正数，例如 5σ）

3.11 直方图规定化

$$s=T(r)=\int_0^r p_r(x)dx=-r^2+2r$$

$$v=G(z)=\int_0^x p_z(x)dx=\begin{cases}2r^2\left(0\le r<\frac{1}{2}\right)\\-2r^2+4r-1\left(\frac{1}{2}\le r\le 1\right)\end{cases}$$

$$G^{-1}(z)=\begin{cases}\sqrt{\frac{z}{2}}\,(0\le z<\frac{1}{2})\\ \frac{2-\sqrt{2-2z}}{2}\,(\frac{1}{2}\le z\le 1)\end{cases}$$

$$z=G^{-1}(s)=G^{-1}\big(T(r)\big)=\begin{cases}\sqrt{\frac{-r^2+2r}{2}}\,(Always)\\ \frac{2-\sqrt{2-2(-r^2+2r)}}{2}\,(Impossible)\end{cases}$$

$=\sqrt{\frac{-r^2+2r}{2}}\,(0\le r\le 1)$

5、夜间道路对比度增强

对于 hw1_dark_road_1.jpg，使用命令：（另外两张图类似）

```
H=imread('hw1_dark_road_1.jpg');
subplot(3,2,1);
imshow(H);
imwrite(H, 'H.jpg');
title('原图');
subplot(3,2,2);
histH = imhist(H);
imwrite(histH, 'histH.jpg');
title('原图直方图');
subplot(3,2,3);
H1=adapthisteq(H);
imshow(H1);
imwrite(H1, 'H1.jpg');
title('adapthisteq 均衡后图');
subplot(3,2,4);
histH1 = imhist(H1);
imwrite(histH1, 'histH1.jpg');
title('adapthisteq 均衡后直方图');
subplot(3,2,5);
H2=histeq(H);
imshow(H2);
imwrite(H2, 'H2.jpg');
title('histeq 均衡后图');
subplot(3,2,6);
histH2 = imhist(H2);
imwrite(histH2, 'histH2.jpg');
title('histeq 均衡后直方图');
```

得到的图放在文件夹“5”的三个子文件夹中。每个子文件夹有 6 张图：原图、adapthisteq 均衡后图、histeq 均衡后图；原图直方图、adapthisteq 均衡后直方图、histeq 均衡后直方图。

a） 原图直方图的灰度级集中在 0～50 处，画面显得很暗。

b） histeq 均衡后图显得过于明亮，以至于天空都变白了，而且灰度级较少，于是尤其是天空很明显是黑一块白一块的。

c） adapthisteq 均衡后直方图的效果好，不仅变明亮了，而且对比度变明显了，全图的效果明暗分明；而且灰度级不会太少，人眼看起来仍然是连续的。

d） 可以直接调用使用 adapthisteq(H1)，也可以填入参数。关于参数选择，我使用的是 adapthisteq(H1,'NumTiles',[8 8],'ClipLimit',0.005);我也尝试过其他参数，比如 adapthisteq(H1,'NumTiles',[8 8],'ClipLimit',0.05); 但是 clip 大了以后，效果就跟 histeq 相接近，尤其是天空很明显是黑一块白一块的。