数字图像处理实验

实验一

直方图均衡、图像增强、反锐化掩模、半色调技术

姓名： 吴侃

学号： 14348134

班别： 2014 级计算机系一班

日期： 2017.03.25-2017.03.27

摘要：

本次实验，我使用 python 编程语言，完成了四个项目，分别为直方图均衡、基于拉普拉斯算子的图像增强、反锐化掩模和半色调技术。除了图像处理方面的函数均为自己编写之外，自己也实现了彩色图像转灰度图像的函数，颜色直方图的显示，以及一个在 python 下计算速度较快地卷积算法。

一、实验环境

编程语言：python 2.7

第三方库：numpy, scipy, matplotlib

二、实验内容

1. Project 03-02

直方图均衡（**Histogram Equalization**） [Histogram-Equalization.py]

直方图均衡化步骤：

记原图像的宽度为 M 像素，长度为 N 像素，灰度级别为 L，颜色 r_k 出现次数为 n_k.

(1) 计算频率

$$p_r(\mathrm{r}_k)=\frac{n_k}{MN},\ k=0,1,2,...,L-1$$

(2) 计算累计分布函数

$$P(r_k)=\sum_{j=0}^{k}p_r(r_j)=\sum_{j=0}^{k}\frac{n_k}{MN},\ k=0,1,2,...,L-1$$

(3) 变换函数

$$s_k=T(r_k)=(L-1)\sum_{j=0}^{k}\frac{n_j}{MN},\ k=0,1,2,...,L-1$$

改进的变换函数：

$$s_k = T(r_k) = [\max(r) - \min(r)]\sum_{j=0}^{k}\frac{n_j}{MN},\ k = 0,1,2,...,L-1$$

改进方法的优势为，能够将灰度值的取值范围限定在原灰度值取值区间内。

(4) 标准化灰度级别

将 s_k 四舍五入转换为标准的灰度级别，限制灰度级别在 0 到 L-1(包含 L-1)内。

2. Project 03-05

使用拉普拉斯算子实现图像增强（Enhancement Using the Laplacian）

[Laplacian-Enhancement.py]

拉普拉斯算子在连续情况下，为二阶函数的二阶梯度：

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

对于离散的图像，拉普拉斯算子的离散形式为：

$$\frac{\partial^2 f}{\partial x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y)$$

$$\frac{\partial^2 f}{\partial y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y)$$

$$\nabla^2 f = [f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1)] - 4f(x,y)$$

拉普拉斯算子对应的卷积核为：

0	1	0	1	1	1
1	−4	1	1	−8	1
0	1	0	1	1	1

这两个卷积核的中心都为负数，左边的为四方向的二阶梯度卷积核，右边为八方

向的二阶梯度卷积核。

卷积核的中心也可以为正数，如图：

0	−1	0	−1	−1	−1
−1	4	−1	−1	8	−1
0	−1	0	−1	−1	−1

使用拉普拉斯卷积核对图像进行卷积后，可以求出原图像各点的二阶梯度。若拉普拉斯滤波中心的系数为负数，用原图像减去二阶梯度；若拉普拉斯滤波中心的系数为正数，用原图像加上二阶梯度。最终得到增强的图像。

$$g(x,y)=\begin{cases} f(x,y)-\nabla^2 f, & \text{当拉普拉斯滤波中心系数为负} \\ f(x,y)+\nabla^2 f, & \text{当拉普拉斯滤波中心系数为正} \end{cases}$$

3. Project 03-06

反锐化掩模（Unsharp Masking）

[High-Boost-Filter.py]

锐化图像 $f_s(x,y)=f(x,y)-\bar{f}(x,y)$

其中 $f(x,y)$ 为原图像，$\bar{f}(x,y)$ 为模糊后的图像。

使用 3x3 的平滑滤波器 $\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$ 对图像 $f(x,y)$ 进行平滑，得到

$\bar{f}(x,y)$

有：$f_{hb}(x,y)=Af(x,y)-\bar{f}(x,y)$

若使用拉普拉斯算子计算锐化图像，有：

$$f_{hb}(x,y) = Af(x,y) - \nabla^2 f(x,y) \quad if \ the \ center \ coefficient \ of \ the \ Laplacian \ mask \ is \ negative$$

$$f_{hb}(x,y) = Af(x,y) + \nabla^2 f(x,y) \quad if \ the \ center \ coefficient \ of \ the \ Laplacian \ mask \ is \ positive$$

4. Project 02-01

基于半色调技术的图像打印程序(Image Printing Program Based on Halftoning]

[halftoning.py]

(1) 缩放图像

由于要让图像在 8.5 x 11 英寸下的纸上显示，假设 DPI 为 96，可计算出分辨率为 816 * 1056. 而在半色调技术下，需要用 3x3 个格子表示原图像的一个像素。因此，为了显示整个图像，需要将原图像缩放到 272 * 352 分辨率以下。

(2) 灰度级划分

原图像的灰度级为 256，而 3x3 的半色调技术只能体现 10 种灰度值。因此将原图像的灰度级除以 25.6，四舍五入后映射到对应的半色调灰度。

5. Python 下的快速卷积的实现

在 python 下，numpy 的内部实现为 C，充分调用 numpy 的批量矩阵计算方法能够提升程序的性能。我实现了一个 python 下的快速卷积方法，该方法尽可能批量进行矩阵计算，并且减少 python 中的循环次数。

卷积的公式：

$$\omega(x,y) \bullet f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x-s, y-t)$$

(1) 初始化

生成一个与原图像 G 大小一致的矩阵 M，其中各个元素的大小为 0；生成一

个从数字到矩阵的映射 D，记录中间结果。

(2) 计算

对于卷积核中的每一个元素 v_{ij}，首先判断 v_{ij} 是否在 D 中存在映射，若存在映射，取出对应的矩阵 H。若不存在映射，另 H=G * v_{ij}，并记录到映射 D 中（从 v_{ij} 到矩阵 H 的映射）。将矩阵 H 的中心移到(i, j)处，然后与矩阵 M 相加。

(3) 结果

最终得到图像 G 与卷积核的卷积结果。

6. 彩色图转灰色图

由人眼视觉模型，得到转换公式：

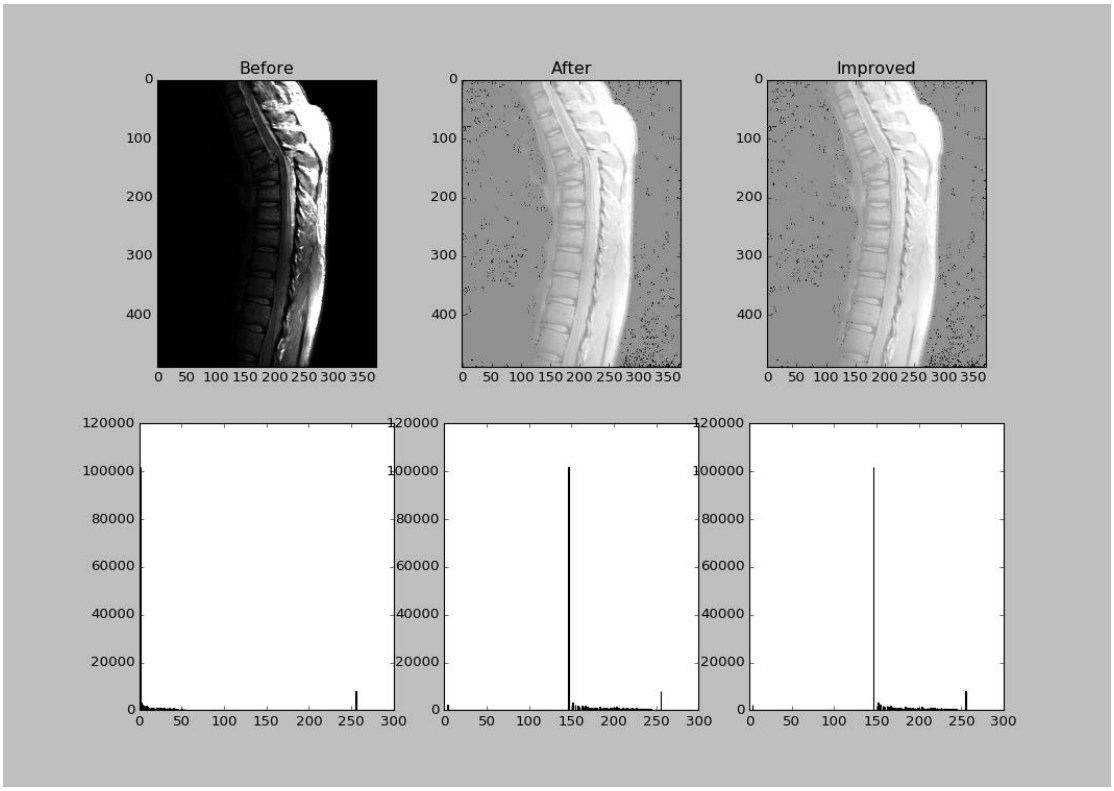
$$G_{gray} = 0.299 * \mathrm{G}_{Red} + 0.587 * \mathrm{G}_{Green} + 0.114 * G_{blue}$$

三、实验结果与分析

1. Project 03-02

直方图均衡（**Histogram Equalization**） [Histogram-Equalization.py]

左一为原图，中间为一般的直方图均衡的结果，右一为改进的直方图均衡的结果。



可以看到，原图像（左一）的灰度直方图中，大部分像素点的颜色靠近灰度值 0。而颜色直方图均衡化后的图像（中间与右一），它们的直方图中，出现次数最多的颜色的灰度值在整个灰度级的中间（灰度值约为 150）。直方图均衡化后的图像（中间与右一）中的物体的左侧，比原图（左一）**更加清晰，整体颜色更白**。在这张图片中，一般直方图均衡的方法与改进直方图均衡的方法得到的新图像的差异不大，原因是测试的图片的灰度值取值范围比较大，max(r)-min(r)接近 L-1.

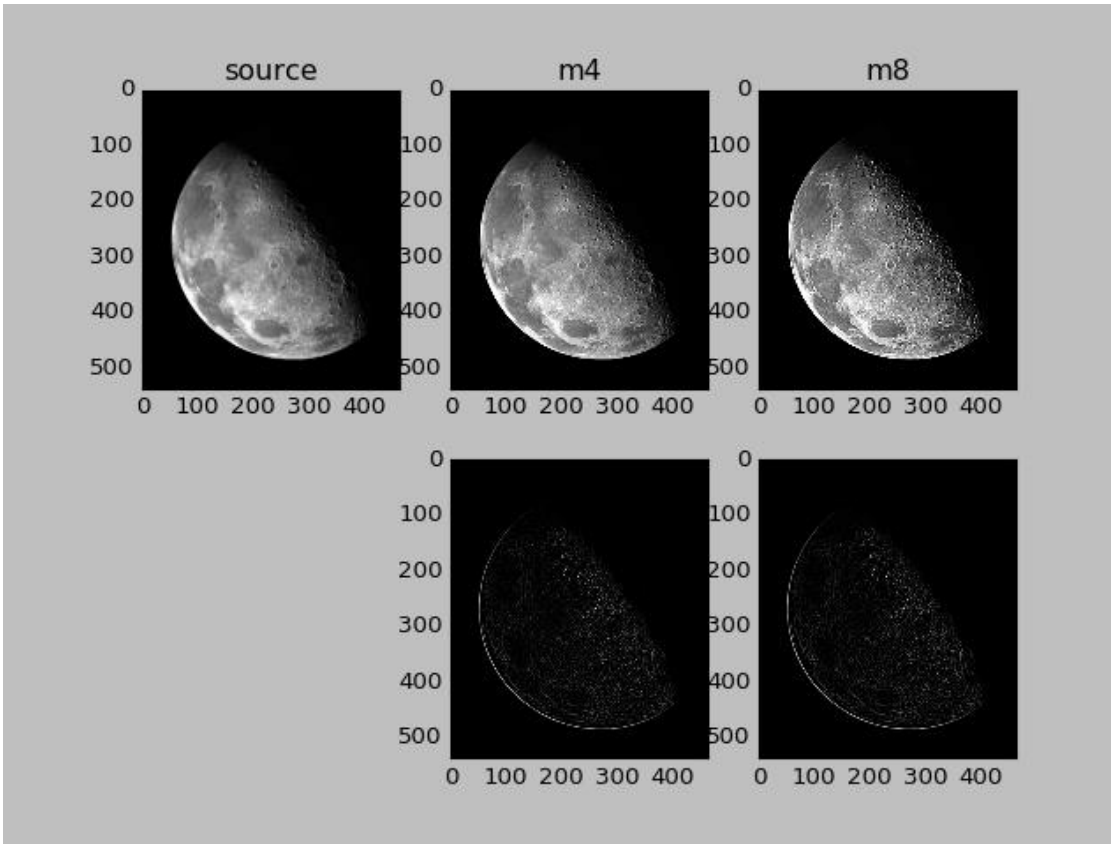
2. Project 03-05

使用拉普拉斯算子实现图像增强（Enhancement Using the Laplacian）

[Laplacian-Enhancement.py]

左一为原图，m4 指使用了四方向的拉普拉斯算子，m8 指使用了八方向的拉普拉斯算子。

其中，上方的中间、右一两图为增强后的图像；下方中间、右一两图为两种算子的二阶微分结果。



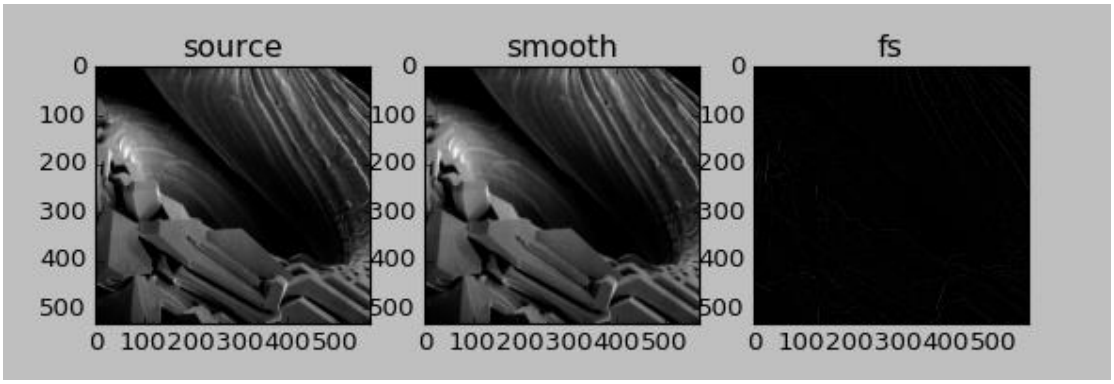
可以看到，拉普拉斯算子对图像的突变有比较强的响应，对灰度变化缓慢的区域响应的值比较小。当将拉普拉斯变换后的图像与原图像叠加，可以保留拉普拉斯锐化处理的效果，又复原了背景的信息。

拉普拉斯变换后的图像（下方的中间与下方的右一）体现了原图的轮廓，增强后的图像（上方的中间与上方的右一）与原图相比，轮廓更加清晰。同时，八方向的拉普拉斯算子得到的结果比四方向的拉普拉斯算子得到的结果，轮廓更加明显，原因是八方向的拉普拉斯算子做了更多方向的微分。

3．Project 03-06

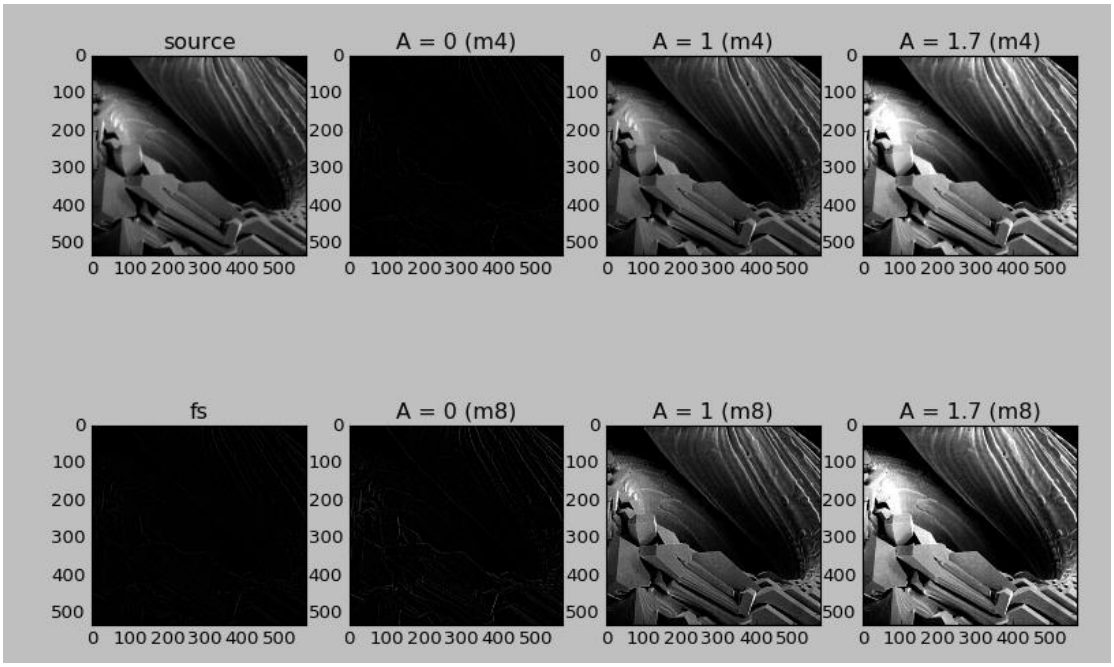
反锐化掩模（Unsharp Masking）[High-Boost-Filter.py]

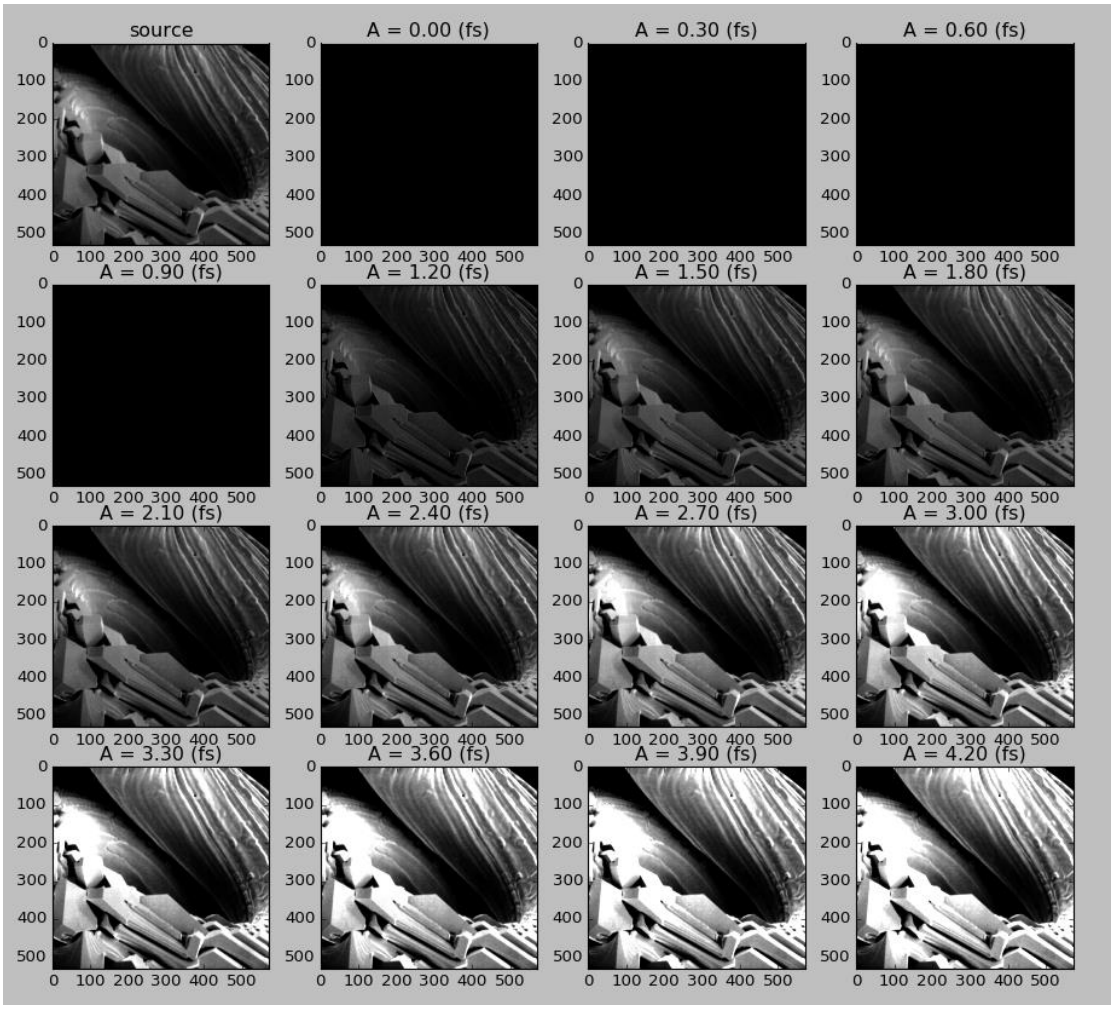
(a) 下图中，左一为原图，中间为平滑后的图像，右一为锐化图像。



由于平滑滤波器的大小为 3x3，模糊的效果不是很明显。但是锐化图像（右一）显示了略微的轮廓。

(b) 分别使用四方向(m4)、八方向(m5)的拉普拉斯掩模作 High-boost 滤波器，以及使用反锐化掩模滤波器(fs)，取不同的 A 值。





各种滤波器得到的锐化图像的清晰程度排序为：

八方向的拉普拉斯滤波器>3x3 的平滑滤波器>四方向的拉普拉斯滤波器

当 A=1 时，High-Boost 滤波器变成了标准的拉普拉斯图像增强。当 A 从 1 逐渐增大时，锐化的作用变得越来越少；当 A 变得足够大时，得到的图像接近于原图像直接乘以一个常数，图像变得越来越亮。

当 A 取值约 1.8 时，得到的结果与 Fig 3.41(d)比较接近。

4．Project 02-01

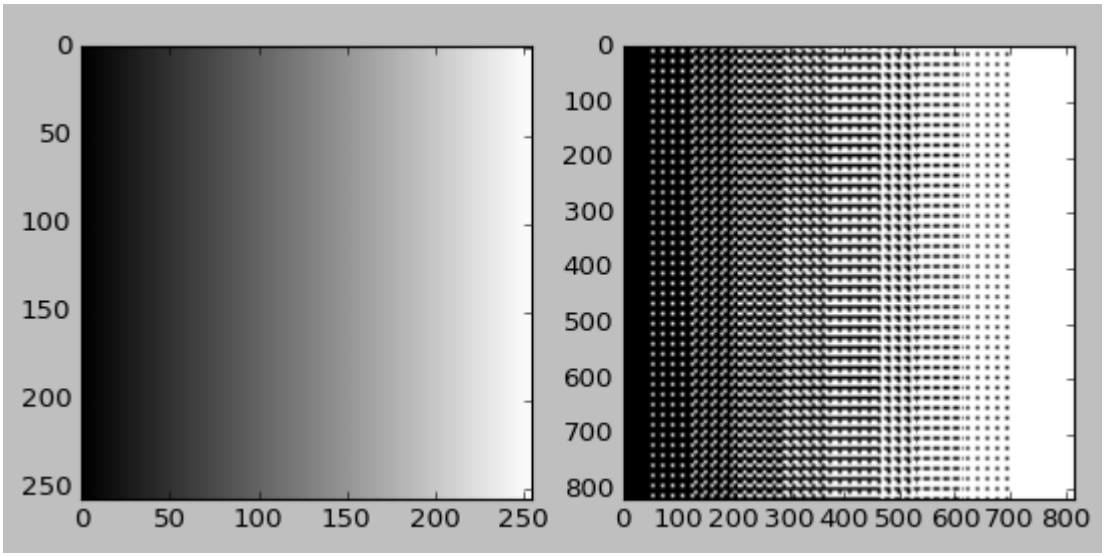
基于半色调技术的图像打印程序(Image Printing Program Based on Halftoning)

[halftoning.py]

注：由于 python 中的 matplotlib 显示图像时，不会改变缩放图像的灰度值（直

接间隔采样)，因此这里也使用了 Windows 自带的图像查看器显示半色调后打印的图像（每组样例的第三张图)。

256 级灰阶图及其半色调打印结果



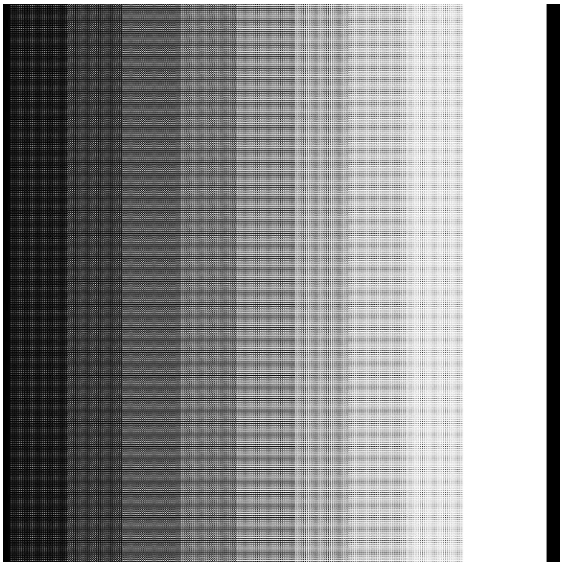


Fig2.22(a)及其半色调打印结果

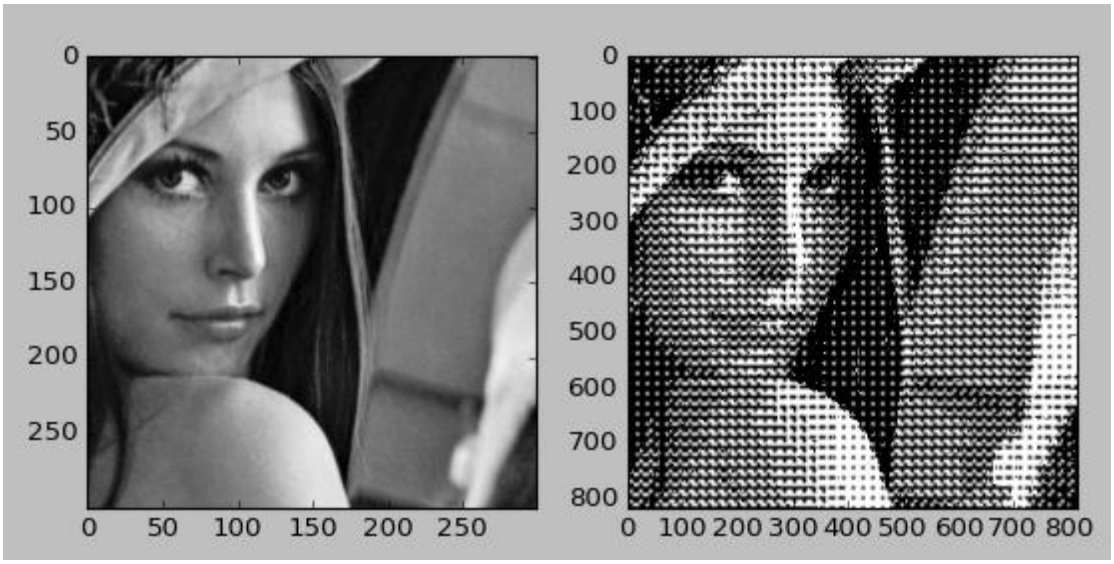


Fig2.22(b)及其半色调打印结果

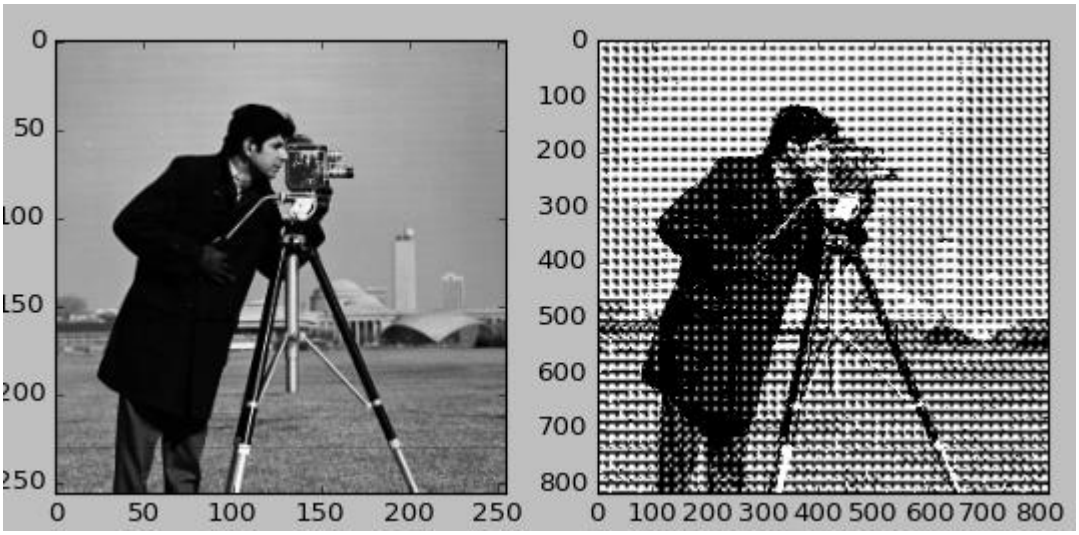
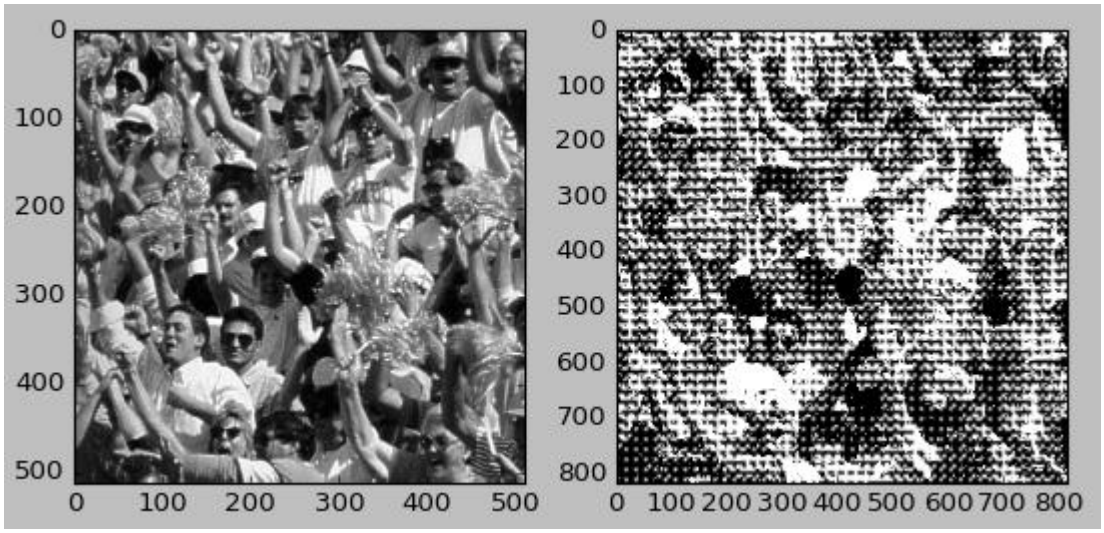


Fig2.22(c)及其半色调打印结果



同意表示等优曲线，在降低像素以及灰度级后，细节较高的图像的细节的模糊程度比细节低的图像的模糊程度要大。原因是细节较强的图片若要达到细节较

弱的图片的同等主观图片质量，需要更高的空间分辨率以及灰度级。