# DIP第二次作业

**软件平台：**

Python3.6+opencv-python4.5.2.54

**实验步骤：**

首先在使用OpenCV中与图像旋转变换有关的函数

M= cv2.getRotationMatrix2D(center, angle, scale )

其中函数的参数部分center为旋转的中心，angle为旋转的角度，scale为旋转的缩放比例。通过此函数来求得旋转过程中的旋转矩阵，在本实验中选取图像的中心点为旋转中心点，第一步选择15度为旋转角度，再次旋转回来时选择345度为旋转角度，旋转的缩放比例为1。

Image = cv2.warpAffine(image, M, (nW, nH),flags = cv2.INTER\_NEAREST)

随后使用opencv中的仿射变换函数来对图像进行变换，其中本函数使用的参数具体为：输入图像、变换矩阵、输出图像大小以及插值的方式。flags参数用于选择不同的插值方式，本次实验用到的插值方法主要有

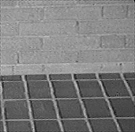
cv2.INTER\_NEAREST （最近邻插值）

cv2.INTER\_LINEAR （双线性插值）

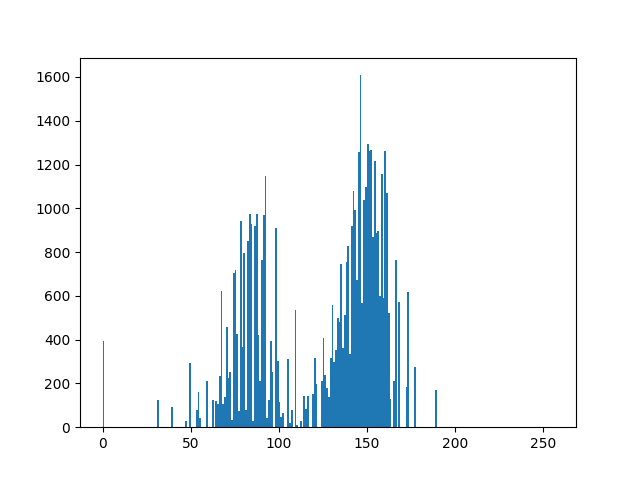
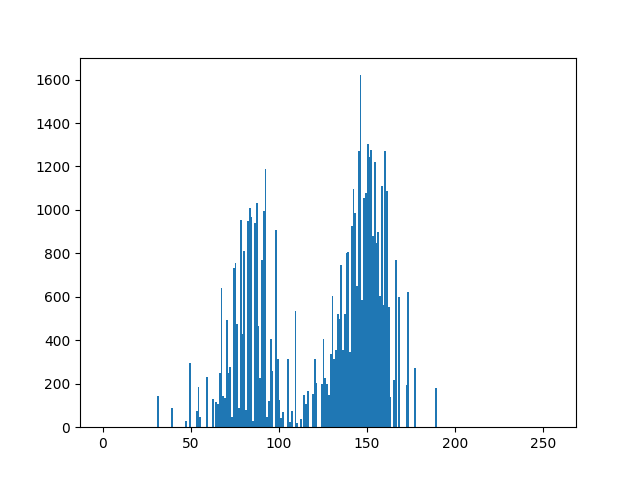
cv2.INTER\_CUBIC （多项式插值）

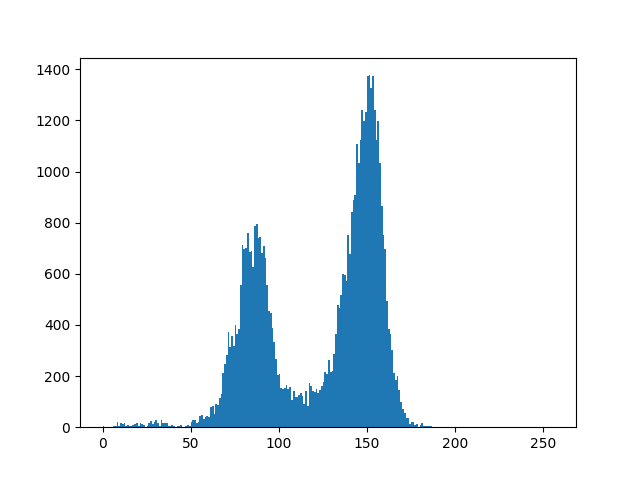
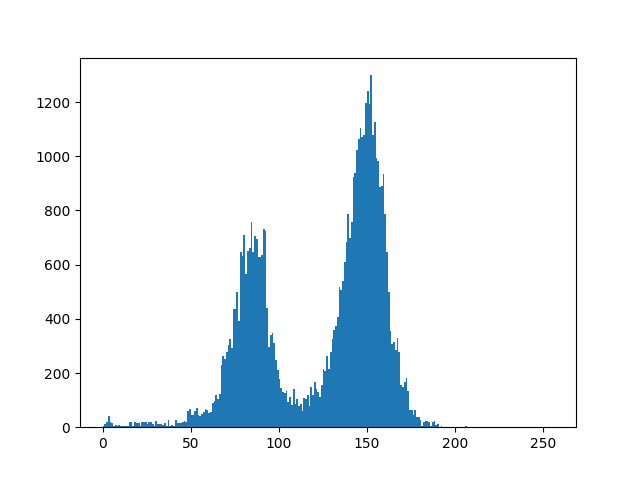
在本实验中需要对正向图像旋转完15度后在进行逆向的旋转15度，在整个过程中图像的大小会随着选择转进行调整，所以最终旋转完成的图像会比原图像大，因此最终需要对旋转后的图像进行裁剪。

几种插值方式得到的最终图像在Pycharm中的显示如下：

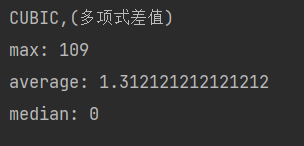
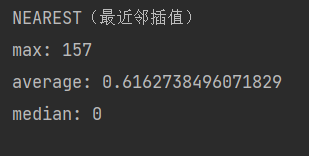
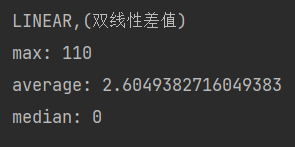
Chapter2\_1 NEAREST LINEAR CUBIC

通过观察可以粗略的看出在旋转完后的图像与原图像并没有发生较大的位移。

 Chapter2\_1 NEAREST

LINEAR CUBIC

通过比较各类插值方法所得到的灰度直方图与原图灰度直方图，可以清晰地看出，在旋转恢复后，图像的灰度分布趋势大致相同，灰阶没有发生改变。

不同插值方法得到的图像与原始图像的误差灰度绝对值的最大值、均值、中值如下表所示：

通过差值后的图像与原图像像素对比中的误差平均值与误差中位数可以看出在插值后的图像与源图像并没有发生位移与像素的偏差。

**结论：**

最近邻插值算法的灰度误差出现了最大值，但它的灰度误差平均值最小，因为它只考虑了最近像素点，而忽略了其他邻域消息，一旦选取了非相关的点插值，尤其是边缘信息的时候，产生的绝对误差是最大的。但它的误差平均值也最小，因此可以认为当图片尺寸较小时，插值的像素数目不多时，大部分值的匹配效果较好，大多数像素点没有误差，因而平均误差小。但由于最邻近差值会使用其相邻像素点的像素值这一特点会导致最终呈现的图像不够平滑，放大后会有锯齿状。双线性插值法的灰度差平均值最大，由于该方法具有低通滤波的性质，导致图像某种程度上变得模糊，与原图像比，出现了较多的差异点。多项式插值法能得到更平滑的边缘，计算精度更高，导致的差异损失也小，因此在数据中就显示平均值中等误差最大值较低的特点。三种插值方法得到的误差灰度中值都是0，说明半数以上的点像素值都与原图的像素值相等。