**17231193 蒋华东**

**实验一 图像变换实验：**

1．实验目的：

学会对图像进行傅立叶等变换，在频谱上对图像进行分析，增进对图像频域上的感性认识，并用图像变换进行压缩。

2．实验内容：

对Lena或cameraman图像进行傅立叶、离散余弦、哈达玛变换。在频域，对比他们的变换后系数矩阵的频谱情况，进一步，通过逆变换观察不同变换下的图像重建质量情况。具体如下：

（1）对图像进行傅立叶变换、获得变换后的系数矩阵；

（2）将傅立叶变换后系数矩阵的频谱用图像输出，观察频谱；

（3）通过设定门限，将系数矩阵中95%的（小值）系数置为0，对图像进行反变换，获得逆变换后图像；

（4）观察逆变换后图像质量，并比较原始图像与逆变后的峰值信噪比（PSNR）。

（5）对输入图像进行离散余弦、哈达玛变换，重复步骤1-5；

（6）比较三种变换的频谱情况、以及逆变换后图像的质量（PSNR）。

3. 实验要求

实验采用获取的图像，为灰度图像，该图像每象素由8比特表示。具体要求如下：

（1）对图像进行傅立叶变换、获得变换后的系数矩阵；

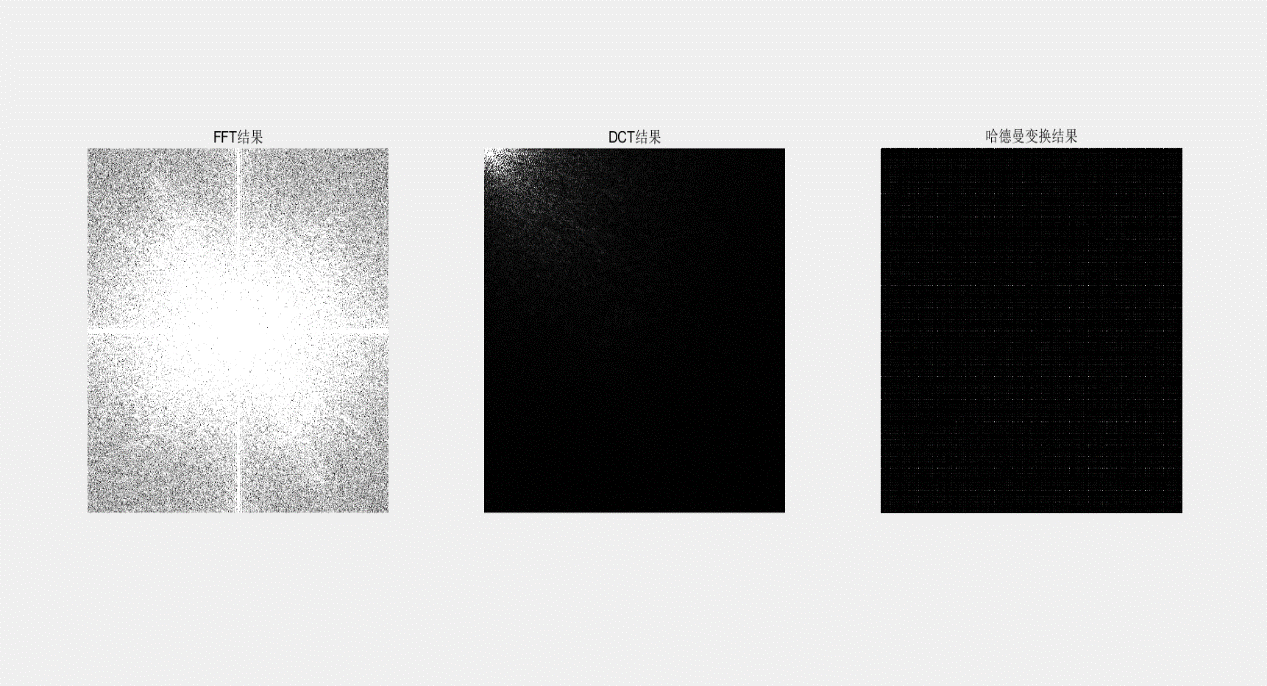
（2）将傅立叶变换后系数矩阵的频谱用图像输出，观察频谱；

（3）通过设定门限，将系数矩阵中90%的（小值）系数置为0，对图像进行反变换，获得逆变换后图像；

（4）观察逆变换后图像质量，并比较原始图像与逆变后的峰值信噪比（PSNR）。

（5）对输入图像进行离散余弦、哈达玛变换，重复步骤1-5；

（6）比较三种变换的频谱情况、以及逆变换后图像的质量（PSNR）。

4．实验结果：

**SNR分别为78.42dB，79.64dB，76.67dB.**

**当设定信噪比为74dB时，需要保留的非零元素大小顺序为HDM<FFT<DCT.**

**具体数值：，需要说明的是由于计算机的配置不高，所以无法把测试间隔设置的较短，所以本实验存在一定的误差。**

4．实验结果分析：

如果小值分布最密集，也就是说黑色连续区域比较大，说明这个图像灰度值接近0的像素点越多；当合理地删除一些像素点的时候，会导致图像压缩所带来的影响产生一定的减小，所以，图像经过逆变换之后反而更加接近于真实值。

从上面的SNR值可以看到：复原效果由大到小排序：DCT>DFT>WHT。

最后从上面的图片可以看到，在要求相同的信噪比的时候，编码效率由大到小大到小排序为DCT>DFT>WHT.

**实验二 反向滤波和维纳滤波：**

1．实验目的：

利用反向滤波和维纳滤波进行降质图像复原，比较不同参数选择对复原结果的影响，增加对图像降质和复原的理解。

2．实验内容：

（1）利用反向滤波方法进行图像复原；

（2）利用维纳滤波方法进行图像复原。

3．实验要求：

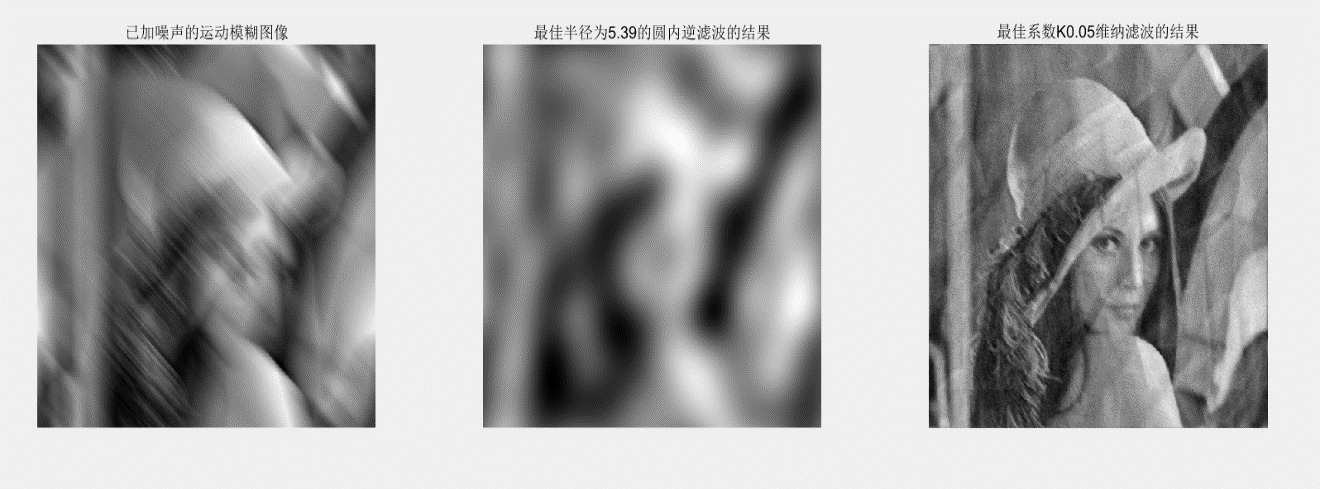
（1）输入图像采用实验1所获取的图像，对输入图像采用运动降质模型，如下式所示：T=5,a=b=0.1. 

（2）对每一种方法通过计算复原出来的图像的峰值信噪比，进行最优参数的选择，包括反向滤波方法中进行复原的区域半径r、维纳方法中的噪声对信号的频谱密度比值K；

（3）将降质图像和利用最优参数恢复后的图像同时显示出来，以便比较。

4．实验结果：





5．实验结果分析：

①对于无噪声运动模糊图像，维纳滤波和逆滤波是一样的，且效果是很好的，当然现实中图像通过信道传输后是不可能不包含噪声的。

②对于有噪声运动模糊图像，逆滤波的取半径滤波效果在PSNR最佳时，图像的效果依然更差，可能是因为选用了PSNR作为了回复效果的参考量。

③对于有噪声运动模糊图像，维纳滤波在K=0.05时取到了最优的PSNR，大约为73dB,但图像并没有回复的很好，原因同上。

**实验三 图像分割处理：**

1．实验目的：

（1）了解图像分割的基本原理，并利用图像分割算法进行图像分割处理。

（2）掌握数学形态学的基本运算。

2．实验内容：

（1）利用类间方差阈值算法实现图像的分割处理；

（2）利用形态学处理进行处理结果修正。

3．实验要求：

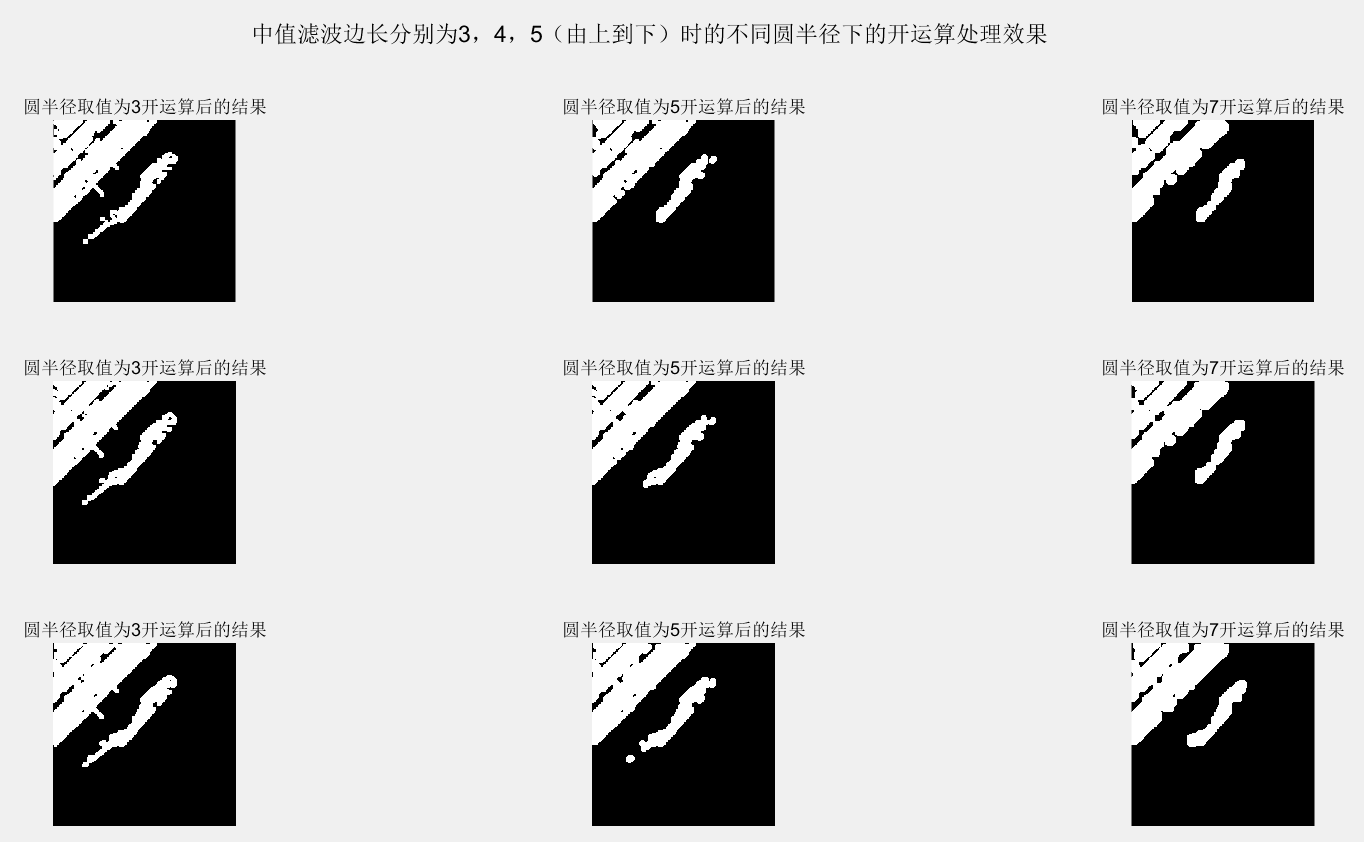
（1）对输入图像进行平滑处理，以减小噪声对分割处理的影响，比较中值滤波范围取不同值时对图像滤波的效果。

（2） 利用类间方差阈值算法对滤波处理后图像进行分割处理，获取分割图像。

（3）利用数学形态学中的腐蚀和膨胀运算处理，剔除分割处理结果中的一些细小的残余误分割点，在进行腐蚀和膨胀运算时可采用半径为r的圆形结构元素，注意比较选取不同r值时的处理结果（r分别取3、5、7）。

4．实验结果：





5．实验结果分析：

①根据第二张实验结果图像，我们可以看出，当选择5\*5滑块进行中值滤波的时候，能消除图像中大部分噪声，且图像结构没有发生畸变，所以我们应该选择5\*5滑块进行滤波。

②从第三张图可以看出，开运算情况下，海面上的噪点并没有消除了，但在选择半径为3的时候，图像结构更加完整。所以，建议选择半径为3的圆结构。

③根据第四张实验结果图像，如果以能过滤掉海绵上大部分白色噪声为评估原则，应该选择半径为7的圆结构.

**实验四 Hough变换：**

1．实验目的：

（1）了解边缘检测算子的原理，并利用边缘算子对图像进行检测；

（2）掌握Hough变换的基本原理。

2．实验内容：

（1）分别将原始图像及加高斯噪声、椒盐噪声后的图像中圆形边缘检测出来；

（2）用Hough变换对边缘进行参数提取。

3. 实验要求：

（1）实验用图像文件：原始图像（houghorg.bmp）、加高斯噪声后图像（houghgau.bmp）和加椒盐噪声后图像（houghsalt.bmp）；

（2）在含有噪声的背景下，先对图像中值滤波，再进行边缘检测；

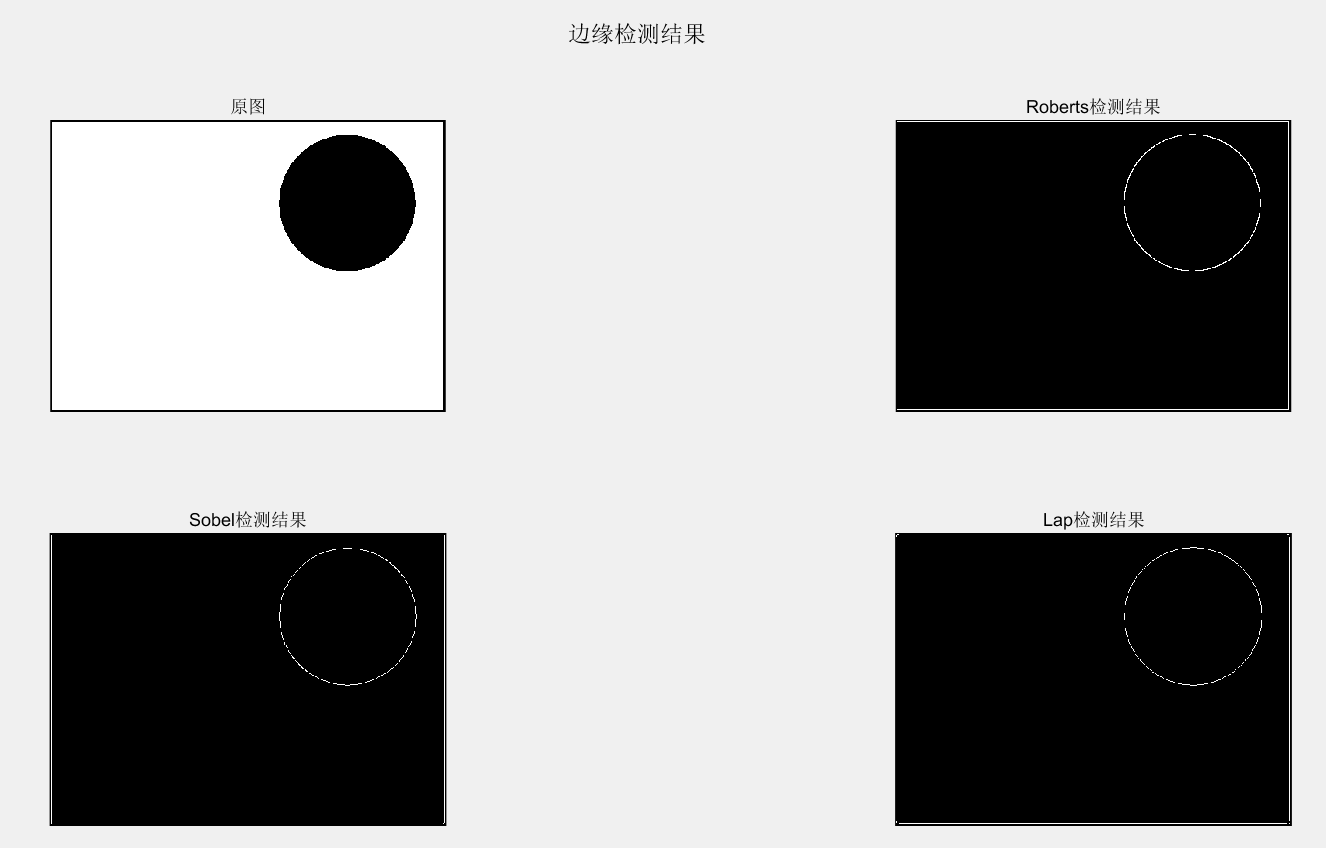
（3）将目标的边界提取出来。边缘检测算子可利用matlab自带函数实现，使用Robert、Sobel和Laplacian算子；

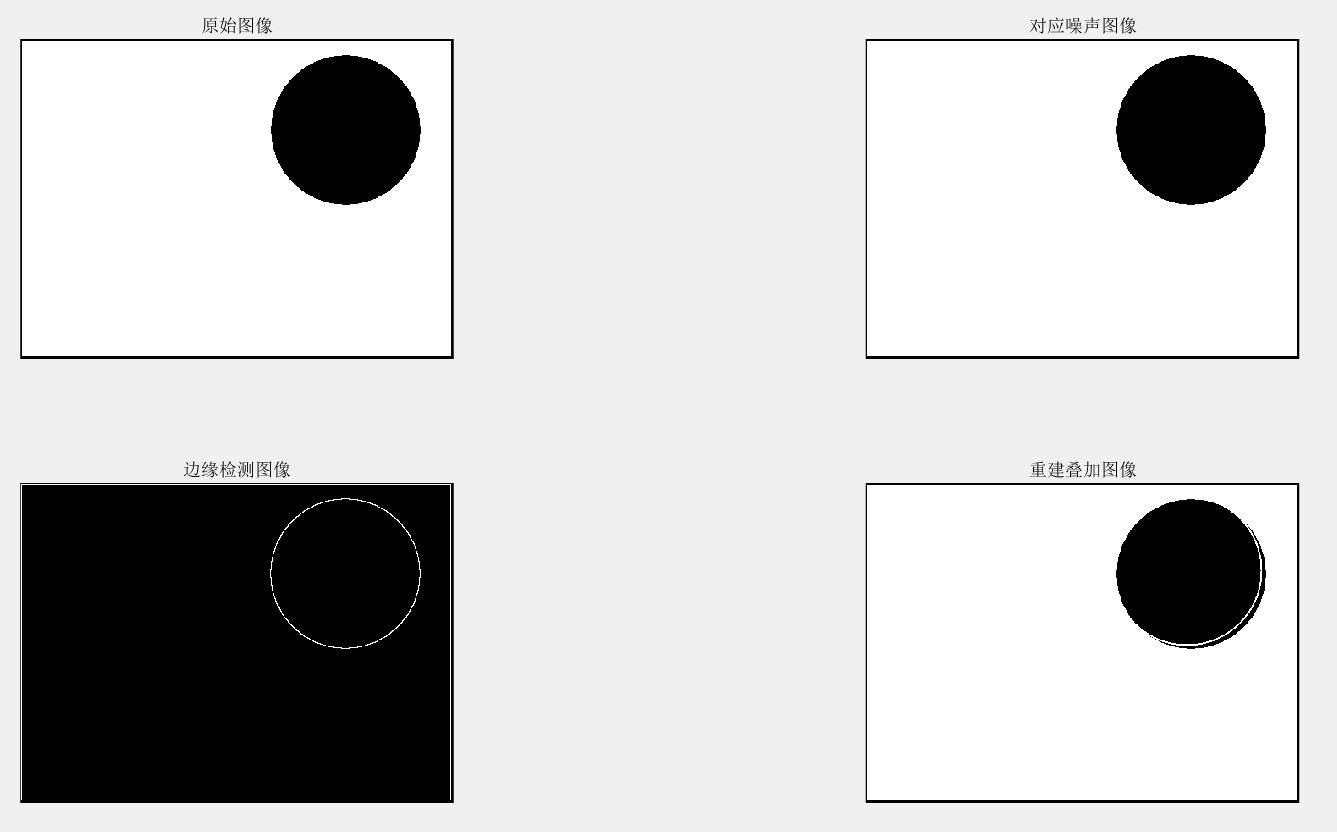
（4）利用Hough变换提取的参数绘制曲线，并叠加在噪声图像上。

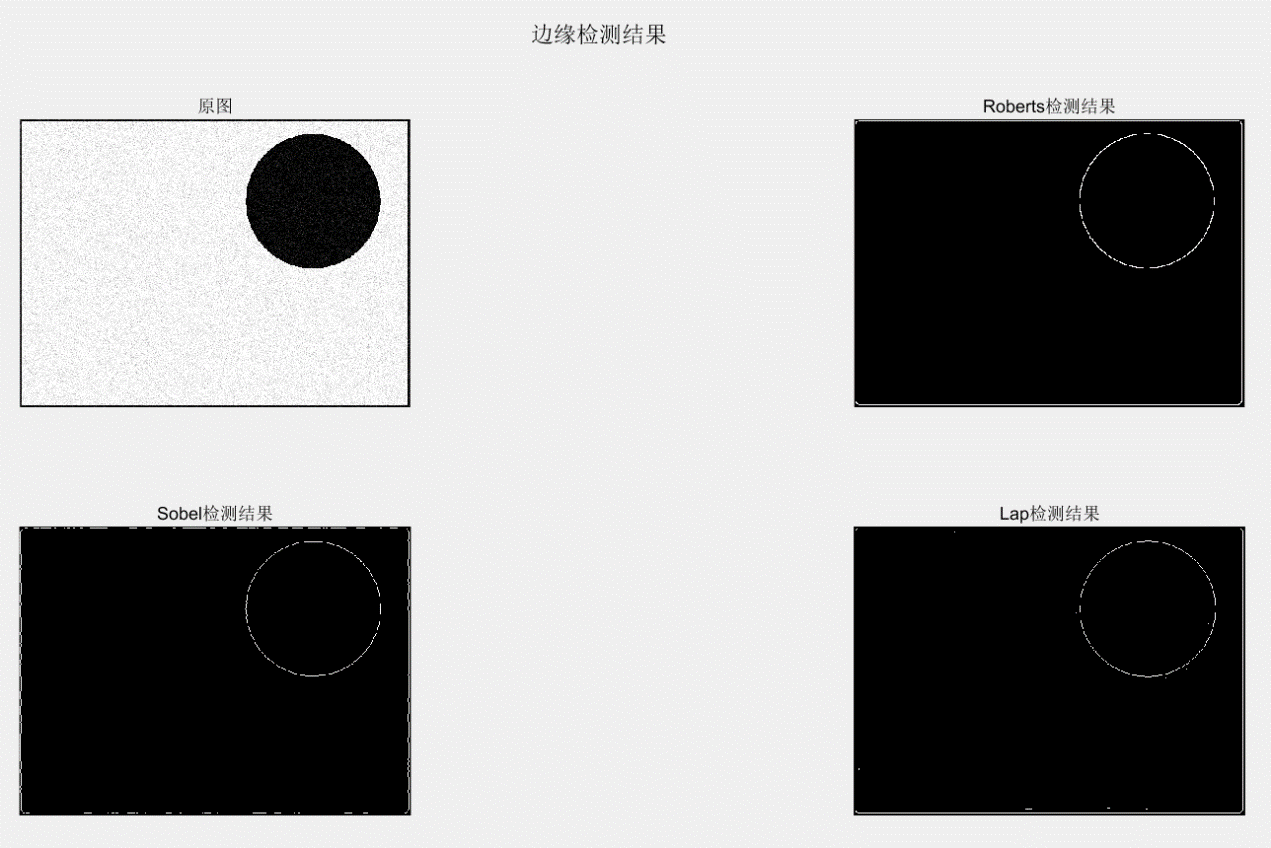
本实验要求用三种不同算子分别提取边缘特征后进行Hough变换提取参数，最好将Hough变换部分单独写成函数。函数的输入包括五个参数：圆半径步长、角度步长、最小圆半径、最大圆半径、投票结果筛选阈值。函数输出为参数矩阵，边界二维图像以及特征域三维矩阵。示例图像中圆半径大概范围为85到90。

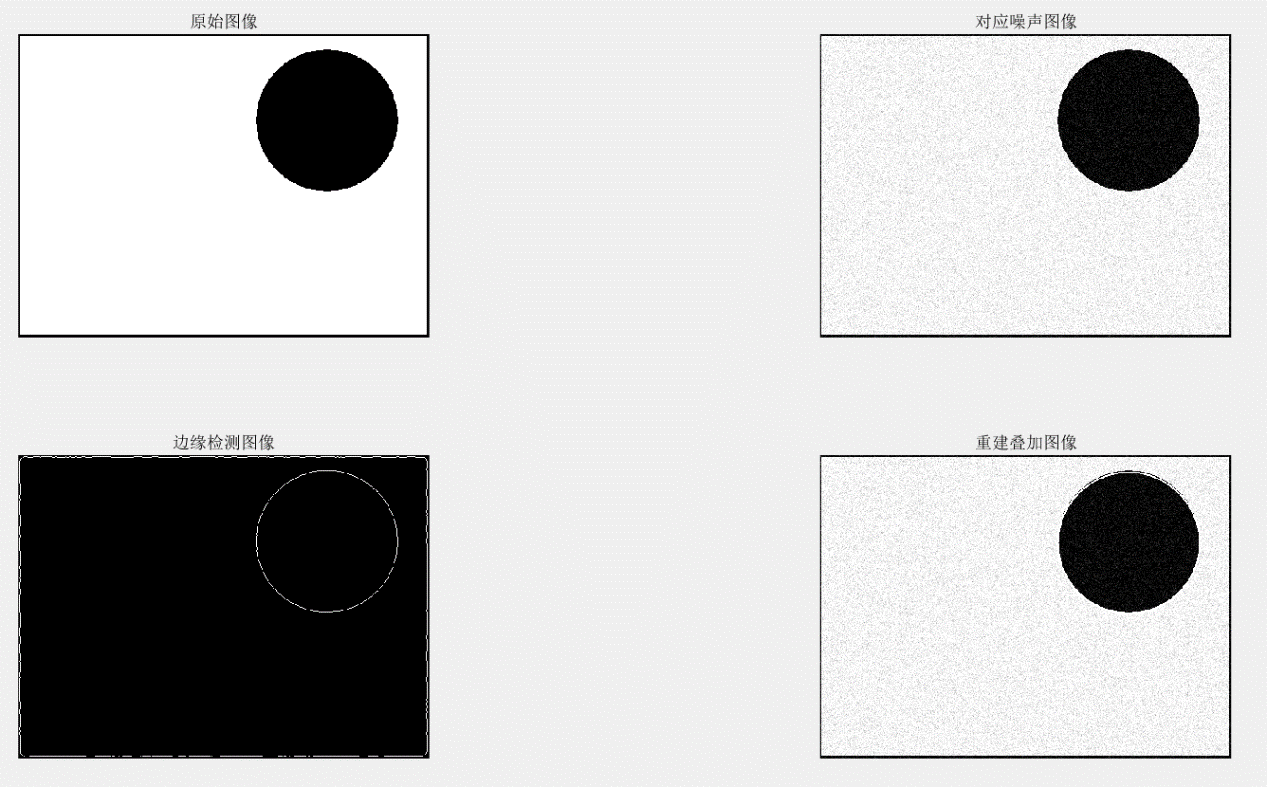
4．实验结果：

（由于图片过多，这里只展示部分结果）









5．实验结果分析：

从实验中我们可以发现Robert边缘检测噪点最少，后续计算量也是最小，也就是说效率高的同时误差也比较小。并且从重建叠加图像可以看出Hough变换与原始图像的边界存在一定的误差。

**实验五 手写字体识别：**

1．实验目的：

掌握分类、识别问题的实质，了解各种分类问题的机器学习方法，并至少掌握一种，熟悉matlab编程。

2．实验内容：

对实验提供的手写数据库（MNIST）进行训练和测试，最终能够较为准确的识别数据库中的手写体数字。

3．实验要求：

编写一完整的Python程序，选取一种合适的机器学习方法，对实验提供的手写数据库（MNIST）进行训练和测试，最终能够较为准确的识别数据库中的手写体数字。

数据文件共分为训练集和测试集：

训练数据集：

Training set images: train-images-idx3-ubyte.gz (9.9 MB, 解压后 47 MB, 包含 60,000 个样本)

Training set labels: train-labels-idx1-ubyte.gz (29 KB, 解压后 60 KB, 包含 60,000 个标签)

测试数据集：

Test set images: t10k-images-idx3-ubyte.gz (1.6 MB, 解压后 7.8 MB, 包含 10,000 个样本)

Test set labels: t10k-labels-idx1-ubyte.gz (5KB, 解压后 10 KB, 包含 10,000 个标签)

数字存储格式：每个数字为28\*28的灰度图，按行拉伸成一个784长的向量以字节形式进行存储。为方便处理，解压后可通过程序读取到NumPy array 中。

测试结果及其格式要求：

1、该算法的精确度，并且不能低于80%；

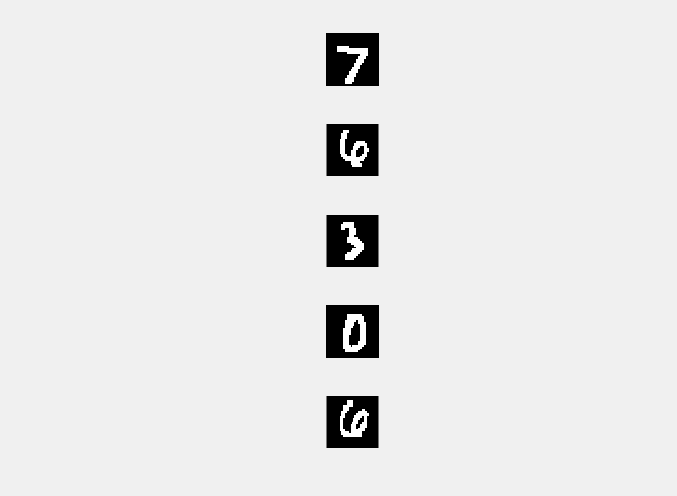
2、有明确的结果统计形式。例：通过图表显示准确率，Loss曲线或测试结果可视化等等。（提示：显示方法可以借助Matlibplot，Tensorboard, Visdom等等）

4. 实验结果

训练测试10次的识别准确率为0.948％。

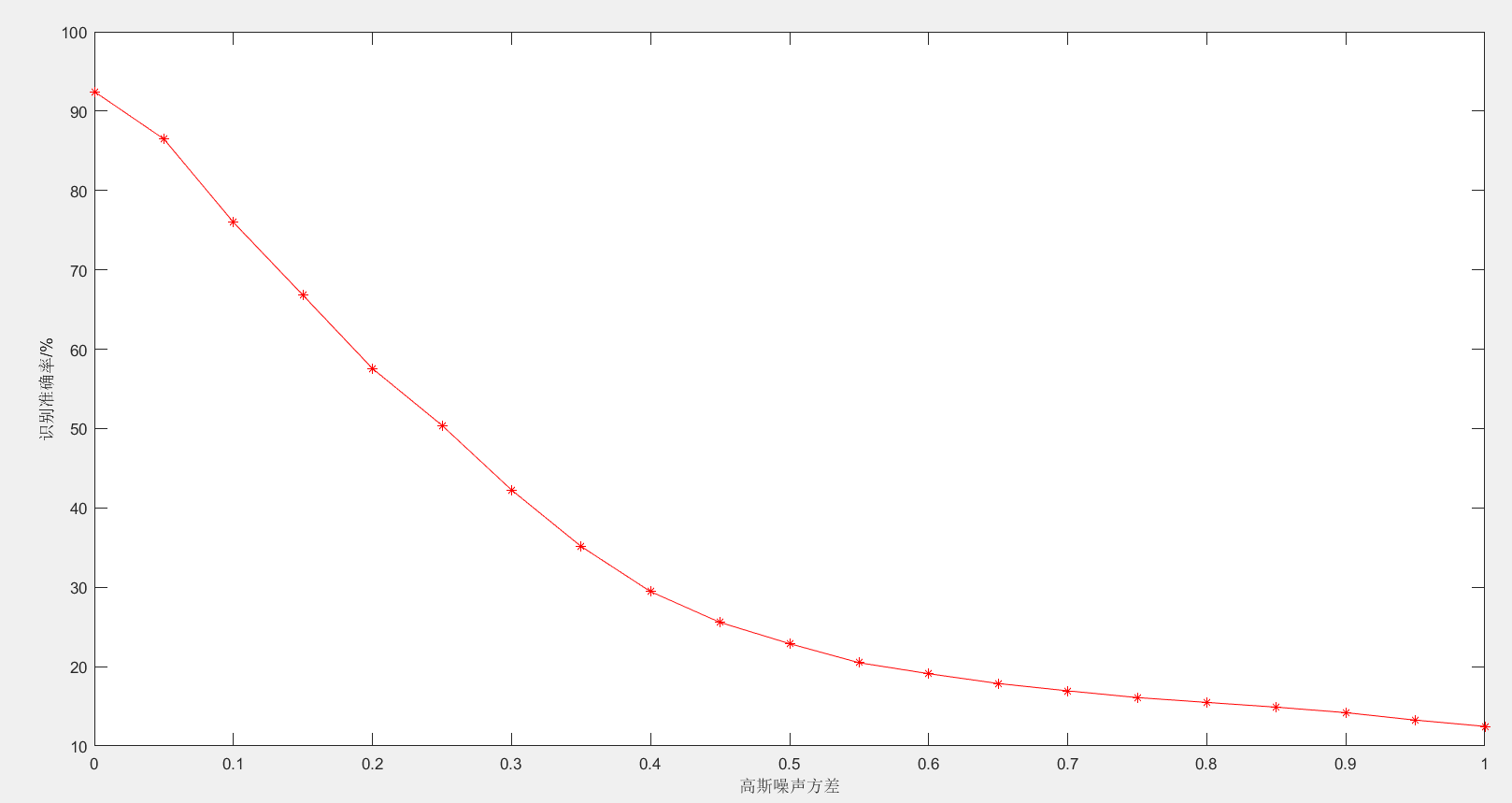
训练测试20次的识别准确率为0.9512%。

训练测试30次的识别准确率为0.9525%。





识别结果中，上一行数字为测试图像中图像的序号（图像内容已经展示时在上一张图片中），下一行则为识别的结果，通过比较两种图片可以发现结果正确。



由上图可以发现噪声对图像识别的准确率影响很大。主要原因应该是训练图像集合之中的图片只有28\*28个像素点，训练模型的抗噪声干扰能力比较弱。

附录

实验一代码

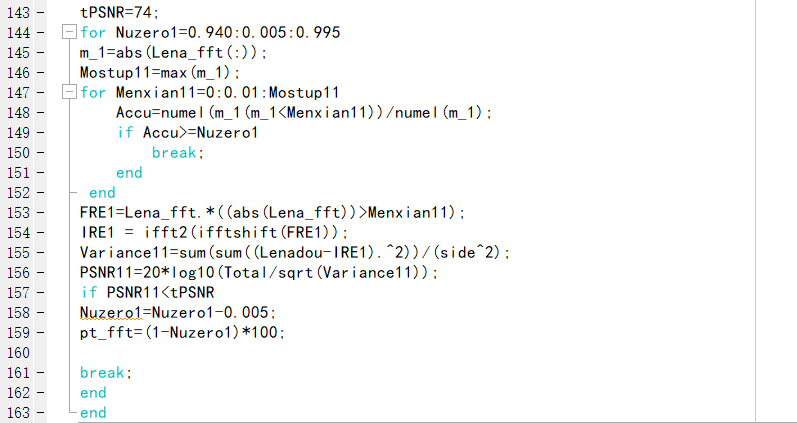
主程序代码

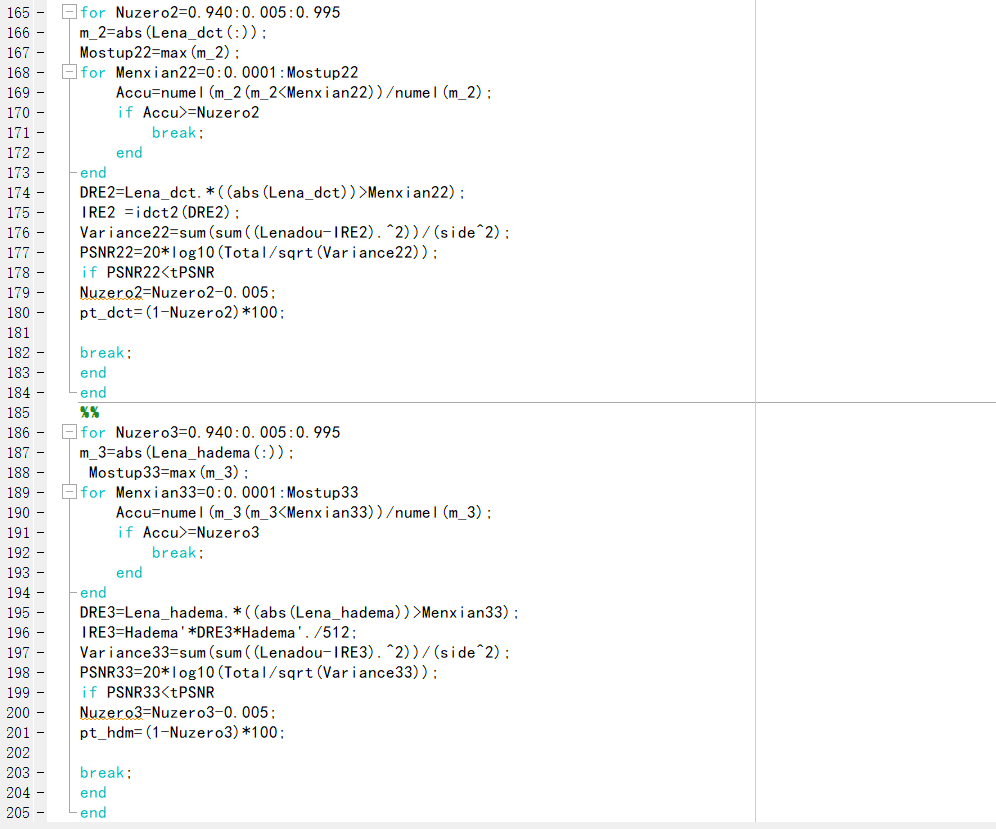












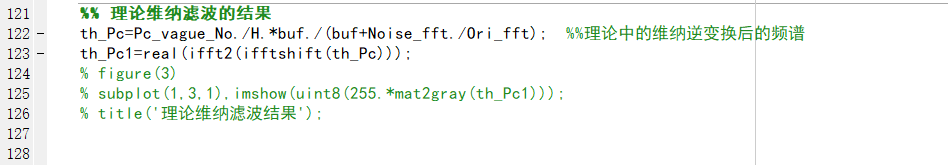
实验二代码

主程序代码

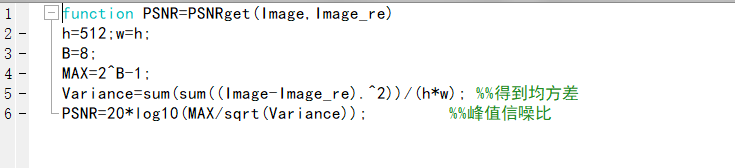








子程序PSNRget代码



实验三代码

主程序代码

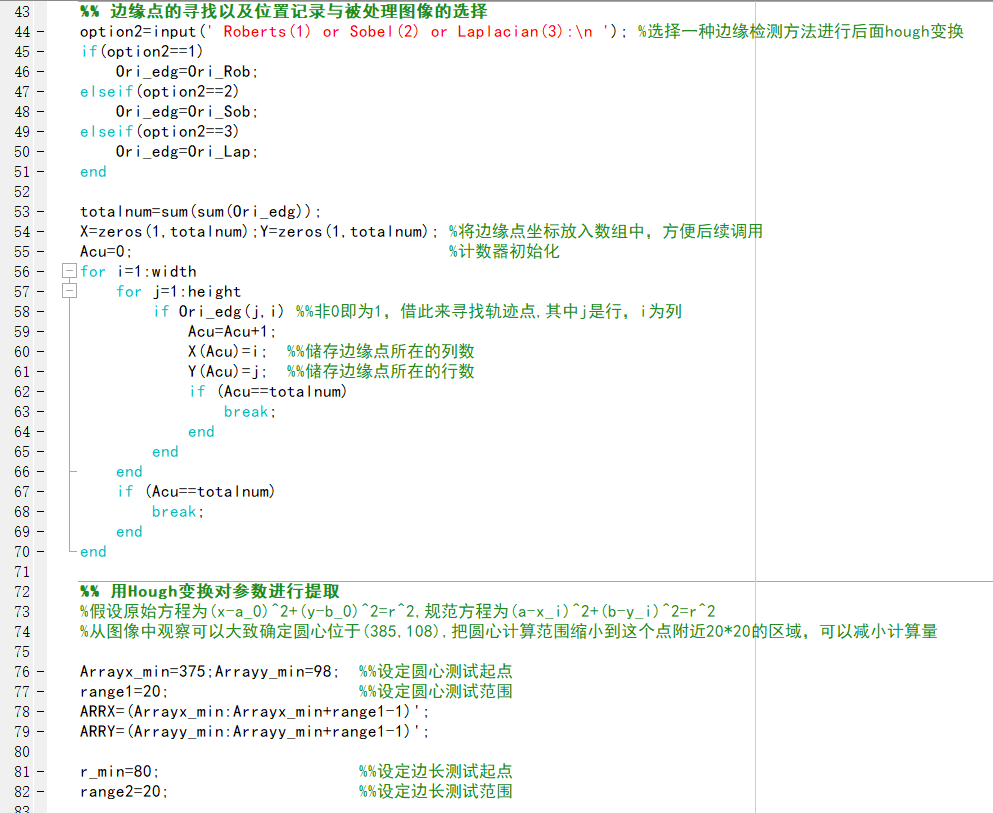




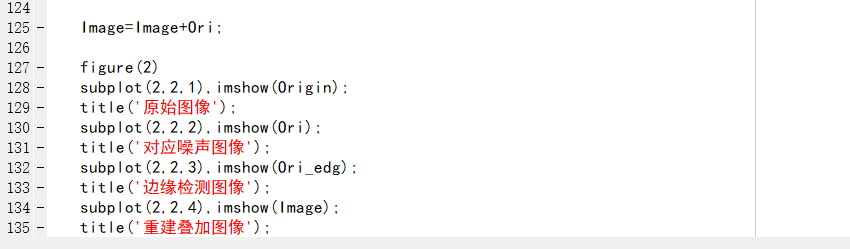
实验四代码

主程序代码









实验五代码

主程序代码

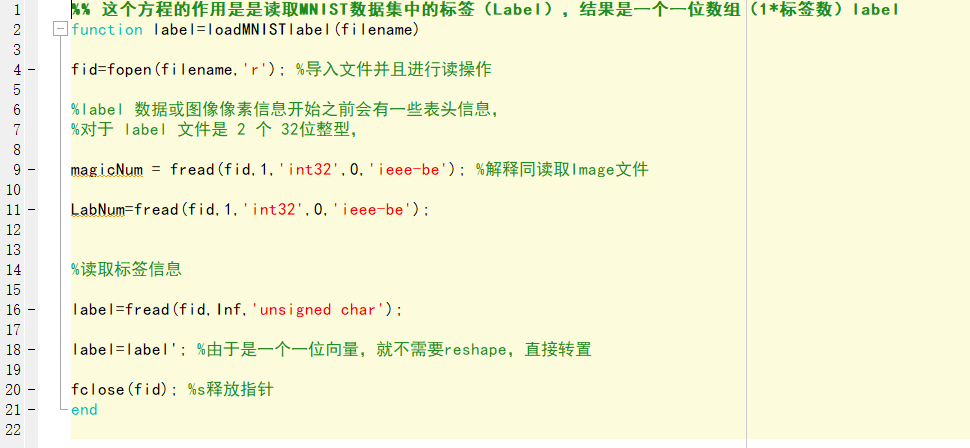




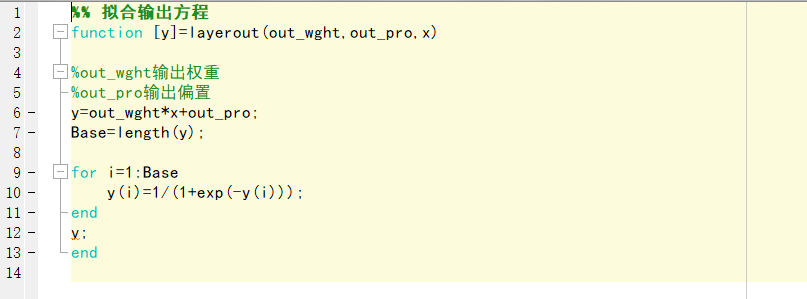
子程序loadMNISTimage代码



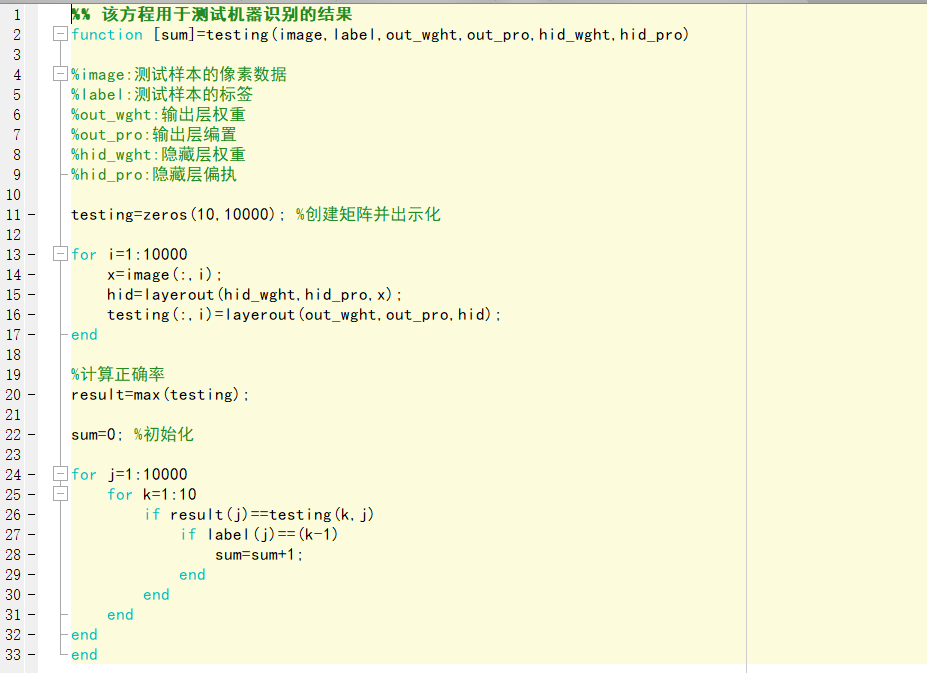
子程序loadMNISTlabel代码



子程序layerout代码



主程序testing代码



子程序Training代码

