## 图像处理与模式识别技术报告

本次大作业主要实现了对于图像的JPEG方式压缩，以及使用两种空间域滤波与一种频率域滤波对于图像进行边缘提取。

**JPEG压缩**

**算法分析**

JPEG算法的基本步骤为：

1. 图像分块处理
2. 离散余弦变换
3. 图像量化处理
4. Zigzag扫描排序
5. 熵编码

**图像分块处理：**

由于在图像量子化过程中，其精度会有损失。在实际编码过程中，为减少损失，将图片划分为若干个8 × 8 的小块，之后在进行后续处理。

**离散余弦变换：**

由于图像经过离散余弦变换后，能量集中于左上角，其余部分数值接近于0，因而经过Huffman等算法进行编码后能够取得较好的压缩效果。

**图像量化处理：**

为了提高压缩效果，可以通过将DCT变换的结果与量化矩阵按元素相除（并取整），以使得结果中有更多的0。

**Zigzag扫描：**

在进行最终的熵编码之前，需要将图像“压”成一维的数组。如果按照自然顺序进行处理，每行的最后一个与下一行的第一个之间联系并不紧密，却会在生成的数组中相邻。而由于相邻的元素之间差距较小，可以选择使用Zigzag的顺序进行扫描。并利用相邻元素间差值较小的特点，将原始数列转换为差值数列进行编码。

**熵编码：**

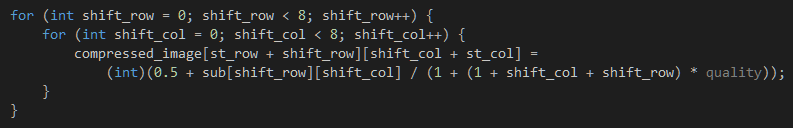
得到差值数列后，使用Huffman编码或算数编码等方式对其进行进一步压缩。

**源码解读**

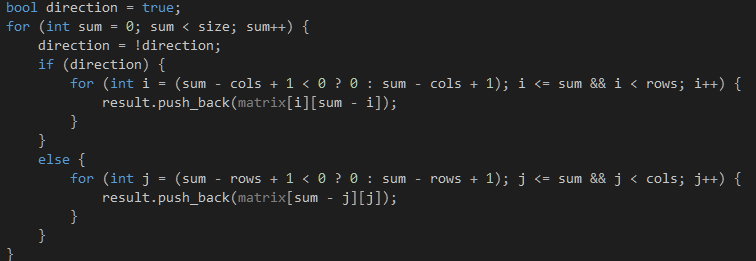
为减少精度损失，首先需要对图像分成8×8的小块，之后再进行DCT变换。在此我们直接调用opencv中Mat结构自带的函数对于相应的行与列进行提取，之后对其进行离散余弦变换。



之后为了进一步提高压缩效果，还要对图像进行量化处理，此处量化矩阵采取动态生成的方式。对于量化矩阵上（i,j）位置的元素，其值为 1 + (1 +shift\_col + shift\_row) \* quality。



Zigzag扫描部分需要注意，扫描延副对角线方向进行，每条对角线上横纵坐标之和相等，并且相邻对角线间和相差一。



之后为了便于压缩，将原始数列转换为差值数列，并利用默认Huffman表对数列进行编码。

**实现效果**

原始图像：



压缩率为4,此时与原始图像差距不大。



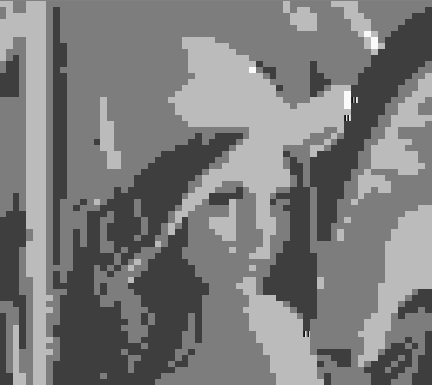
压缩率为10，此时可以发现，图像已经出现了轻微的模糊。



压缩率为50，此时图像受损已经比较严重，出现比较明显的模糊。



压缩率为500，图像受损相当严重，失去了大部分的细节信息。



**图像边缘提取**

**算法分析**

总的来说，图像边缘提取的方式主要有两种：空间域滤波与频率域滤波，此种划分方式主要关注提取操作所作用的域。

**空间域滤波**

空间域滤波直接对图像本身进行操作。具体来说，又分为一阶微分与二阶拉普拉斯微分。此方法使用微分操作完成边缘提取任务，将边缘看作是图像上变换率较大的区域，因而经过求导之后其导数也较大；另一方面，非边沿区域，其变化率则较小。从而在处理过的图像中，边缘显示为较亮的区域，非边缘区域显示为较暗的区域。

而具体说来，微分又分为一阶微分与二阶微分。图像中的一阶微分是用梯度幅值来实现的。对于函数f(x,y)，f在坐标(x,y)处的梯度定义为二维列向量。



它指出了在位置（x，y）处的f的变化率最大的方向。

梯度向量的幅度值表示为M（x,y），即



在实际实现中，往往使用绝对值来近似平方和与平方根操作以提高计算效率。

由于数字图像都是离散的，我们需要对于上述的公式定义一个近似离散，即找出方法来定义gx与gy。在早期数字图像中，往往使用Roberts交叉微分来近似。对于一个以z5为中心的一个3×3邻域对gx和gy的近似如下式：





3×3邻域示意图如下：

|  |  |  |
| --- | --- | --- |
| Z1 | Z2 | Z3 |
| Z4 | Z5 | Z6 |
| Z7 | Z8 | Z9 |

一阶微分计算某坐标处梯度方向的模作为该位置的值，因此，每当像素值发生可观的变化，经过一阶微分后，该位置即显示为亮点，也即标记为边缘。然而，需要注意的是，数字图像的边缘在灰度上常常类似于斜坡过渡（即边缘本身有宽度，并且在宽度范围内亮度仍在平缓变化），这样就导致图像的一阶微分产生较粗的边缘。当处理较精细的图像时，一阶微分往往效果并不好。这时我们需要使用二阶微分，二阶微分强调的是图像的“突变”，即变化的变化，而非灰度级缓慢变化的区域。在实践中，最常使用的是拉普拉斯算子。一个二维图像的拉普拉斯算子定义为



其离散形式为





除空间域滤波外，在频率域的滤波也是经常使用的方法。频率域滤波即使用傅里叶变换，将图像从空间域转换到频率域。而在频率域中，低频信号用于拟合整体（变化较小的部分），高频信号用于拟合细节（变化较大的部分）。因此，我们可以将图像的高频分量滤出，并进行逆傅里叶变换转换回空间域，从而得到图像边缘。

实际中，主要使用的高通滤波器有三种，理想高通滤波器（IHPF），巴特沃斯高通滤波器（BHPF），高斯高通滤波器（GHPF），在上一次大作业中已经详述过，此处只给出函数形式：

**IHPF**



**BHPF**



**GHPF**



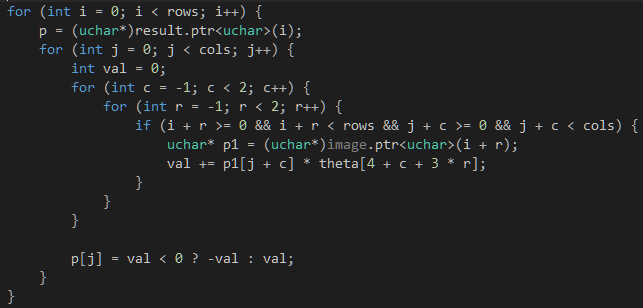
实现时，只需将频率域（u,v）位置的数值与滤波器H(u,v)相乘即可得到滤波后的频率域表示。

**源码解读**

对于一阶与二阶微分，可以使用用算子矩阵与空间域图像按位相乘并求和的方式实现，此处以拉普拉斯算子为例。首先定义算子矩阵（一维化压成数组）：



之后对于（i,j）位置的点，将在原图像上，其左上，正上，右上，左，中，右，左下，下，右下的像素点值分别与算子矩阵的对应点相乘，并将其结果加和，作为该点的最终结果。



对于频率域滤波，首先对其进行傅里叶变换，之后进行fft\_shift处理，以便完成某种高频滤波操作，最终对滤波的结果进行逆傅里叶变换，得到滤波后的图像，具体实现细节已经在上一次大作业中给出，此处只给出具体代码：

**BHPF**



此处实现了一个高频加强器，偏移为a，增强系数为b。

**GLPF**



变量的定义与之前相同，简单实现了GLPF公式

**GHPF**



此处实现了一个高频加强器，偏移为a，增强系数为b。

original\_mat为原始频率域表示，result为处理后的表示。

**实现效果**

原始图像



一阶微分（Roberts算子）



由上图可以看出，一阶微分确实可以完成边缘提取的任务。然而由于其提取每个亮度值发生变化的位置，而边缘的内部亮度往往也是平缓变换的，所以边缘提取的较粗。在此图中，Lena的头发部分由于细节较多，边缘提取较粗，提取效果比较差。

二阶微分（拉普拉斯算子）



二阶微分只提取亮度发生“突变”的位置，即亮度的变化率发生变化的区域，这一版只会在边缘的边缘发生。因此二阶微分提取的边缘基本上就是两条线，效果很精细。然而由于边缘过细，提取效果看上去并不如一阶微分明显。

频率域滤波（理想高通滤波器）



频率域滤波（巴特沃斯高通滤波器）



频率域滤波（高斯高通滤波器）



由上面三张图，使用高通滤波器也可以实现边缘提取。然而由于并不是严格按照亮度变化率进行提取，滤波器的提取结果给人感觉线条更加“柔和”，更接近原图像，与其说是边缘提取，更像是勾勒出原图的轮廓。

**总结**

总的来说，一阶空间滤波提取出的边缘比较粗，直观上最为明显，然而也因此无法处理细节较多的图像；二阶空间滤波将边缘提取为两条线，最为精细，但不进行后续处理，看起来边缘不是很明显；频率域滤波也可以完成边缘提取的任务，但提取出的边缘相较前两种更为“柔和”，更接近原图，更适合提取边缘进行图像增强的任务，而图像划分或者实体识别的工作，更适合使用前两种提取方法。