**实验4 数字图像编码实验**

本次作业需要提交以下内容：

|  |  |
| --- | --- |
| 提交内容 | 详细要求 |
| 作业文档 | 对算法原理进行简单说明；  展示实验效果（每个实验的测试图片不得少于5张，实验用图像自行选择。）；  对实验结果进行分析。 |
| 程序源代码 | 相关程序的全部源代码，要求能够正常编译和运行。 |

**作业一: 无损编码/压缩算法实验**

问题1: 实现行程编码压缩, 肉眼观察压缩效果，并计算原图和压缩以后的尺寸，计算压缩率并比较分析；

|  |  |  |  |
| --- | --- | --- | --- |
| 输入图像 | 原图内存占用（KB） | 压缩图内存占用（KB） | 内存压缩比 |
|  | 703.22 | 1182.29 | 168.13% |
|  | 768.09 | 1021.99 | 133.06% |
|  | 223.27 | 262.90 | 117.75% |
|  | 768.09 | 1403.55 | 182.73% |
|  | 669.64 | 1274.19 | 190.28% |

实验分析：

在本实验室中，我使用的是之字形的2D图像的行程压缩算法。

我们发现在测试样本中，最佳的行程压缩结果也是原图像大小的117.75%倍的大小，通过对比观察五幅实验图片，我们推断是由于图像的颜色空间的连续单调度决定的，如果图像的色素块之间越是连续单调，则压缩的效果越好。我们相信，如果颜色足够连续单调，那么行程压缩的效果肯定是很好的。

问题2: 实现哈夫曼压缩, 肉眼观察压缩效果，并计算原图和压缩以后的尺寸，计算压缩率并比较分析；

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 输入图像 | 原图内存占用（KB） | 压缩后的编码占用（KB） | 压缩后的字典占用（KB） | 压缩图内存占用（KB） | 内存压缩比 |
|  | 703.22 | 0.09 | 13.78 | 13.88 | 1.97% |
|  | 768.09 | 0.09 | 4.54 | 4.63 | 0.60% |
|  | 223.27 | 0.09 | 13.78 | 13.88 | 6.21% |
|  | 768.09 | 0.09 | 27.30 | 27.40 | 3.57% |
|  | 669.64 | 0.09 | 27.30 | 27.40 | 4.09% |

实验分析：

在本实验中，我先对原图按照一个字节为单位进行分割，构建哈夫曼树，然后编码，用编码之后的字符来代替原字节。并且，由于哈夫曼编码的特性，我可以使用变长字节来对新的编码字符进行存储，而不需要再固定每个字节8个长度单位。这样就可以节省很多的空间。可以通过上表观察到重新进行变长编码后的编码空间为原空间的0.01% 。

而保存哈夫曼字典仍然需要占据一定较大的空间，而这部分空间是编码后的压缩图片中占主要空间消耗。

问题3: 实现一维无损预测压缩, 肉眼观察压缩效果，并计算原图和压缩以后的尺寸，计算压缩率并比较分析.

|  |  |  |  |
| --- | --- | --- | --- |
| 输入图像 | 原图内存占用（KB） | 压缩图内存占用（KB） | 内存压缩比 |
|  | 703.22 | 0.09 | 0.01% |
|  | 768.09 | 0.09 | 0.01% |
|  | 223.27 | 0.09 | 0.04% |
|  | 768.09 | 0.09 | 0.01% |
|  | 669.64 | 0.09 | 0.01% |

实验分析：

在本实验中，我通过人为定义了一个线性预测函数，来对图像中的下一个像素值继续递推预测。为了进一步提高预测编码的效率，我先对图像进行了行程编码处理，在行程编码之后再进行预测编码。

由于使用预测编码， 特别是在存在大量的连续像素空间的图片中，可以很有效率的节省像素的存储位数，这样可以大幅减少存储空间。

最后，我对修正矩阵和预测函数的线性参数进行了保存，得到的非常好的无损压缩效果。

**作业二: 有损压缩/压缩算法实验**

查阅JPEG编码的有关资料，对图像进行JPEG压缩，算法步骤必须包括如下几个部分：图像分块，离散余弦变换，量化，ac和dc系数的Z字形编排。

问题1: 质量因子分别选为20，60，80，对比显示原图与不同质量因子下解码后的图像；

问题2: 记录图像大小、压缩比、均方根误差；对结果进行分析。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 输入图像（质量因子 = 20） | 图像大小（KB） | 压缩后大小（KB） | 压缩比 | 均方根误差 | 还原图像 |
|  | 708.84 | 0.09 | 0.01% | 105.91 |  |
|  | 768.09 | 0.09 | 0.01% | 107.81 |  |
|  | 229.78 | 0.09 | 0.04% | 83.38 |  |
|  | 768.09 | 0.09 | 0.01% | 111.73 |  |
|  | 679.97 | 0.09 | 0.01% | 101.73 |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 输入图像（质量因子 = 60） | 图像大小（KB） | 压缩后大小（KB） | 压缩比 | 均方根误差 | 还原图像 |
|  | 708.84 | 0.09 | 0.01% | 105.57 |  |
|  | 768.09 | 0.09 | 0.01% | 109.28 |  |
|  | 229.78 | 0.09 | 0.04% | 84.53 |  |
|  | 768.09 | 0.09 | 0.01% | 111.51 |  |
|  | 679.97 | 0.09 | 0.01% | 104.22 |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 输入图像（质量因子 = 80） | 图像大小（KB） | 压缩后大小（KB） | 压缩比 | 均方根误差 | 还原图像 |
|  | 708.84 | 0.09 | 0.01% | 105.56 |  |
|  | 768.09 | 0.09 | 0.01% | 108.09 |  |
|  | 229.78 | 0.09 | 0.04% | 98.42 |  |
|  | 768.09 | 0.09 | 0.01% | 110.57 |  |
|  | 679.97 | 0.09 | 0.01% | 105.58 |  |

实验分析：

在本实验中，我的实验流程是这样的：

1. 将图像从RGB表示转化为YCrCb表示。
2. 对图像进行 Padding ，和8\*8的切块。
3. 对图像进行离散余弦变化DCT。
4. 对DCT之后的结果使用指定的质量因子进行量化。
5. 对量化之后的频谱图进行行程编码，得到一维数组。
6. 对行程编码后的结果进行霍夫曼变化，进一步合并和减少内存空间。
7. 对霍夫曼结果进行还原。
8. 进行量化还原。
9. 进行IDCT离散余弦变化的逆变换。
10. 将图像从YCrCb空间重新转回RGB空间，进行输出。

通过调整量化的质量因子，会影响图像的量化结果，质量因子越大，图像量化的结果越粗糙，栅格化效果和噪声越严重，图像信息损失的也越多。但是相应的好处是可以进一步减少压缩图的存储空间。

人为定义了一个线性预测函数，来对图像中的下一个像素值继续递推预测。为了进一步提高预测编码的效率，我先对图像进行了行程编码处理，在行程编码之后再进行预测编码。

由于使用预测编码， 特别是在存在大量的连续像素空间的图片中，可以很有效率的节省像素的存储位数，这样可以大幅减少存储空间。

最后，我对修正矩阵和预测函数的线性参数进行了保存，得到的非常好的无损压缩效果。

但，通过实验我感觉质量因子大于20，其实意义不大，图像的质量损失实在是太严重了。