华 中 科 技 大 学

网络空间安全学院

本科：《多媒体数据安全》实验报告

姓　　名 _____

班　　级 _____

学　　号 _____

联系方式 _____

分　　数 _____

评 分 人 _____

实验报告及代码和设计评分细则

评 分 项 目		满分	得分	备注
图像 LSB 隐写与分析方法	实验概述	5		
	详细设计	10		
	代码实现	10		
	测试及结果分析	10		
图像 jsteg、f3、f4 隐写方法	实验概述	5		
	详细设计	10		
	代码实现	10		
	测试及结果分析	10		
文档格式（段落、行间距、缩进、图表、编号等）		10		
感想（含思政）		10		
意见和建议		10		
实验报告总分		100		
教师签名			日　　期	

目　录

实验一 图像 LSB 隐写与分析方法

一、 实验一概述

1.1 实验名称

图像 LSB 隐写与分析方法

1.2 实验目的

掌握图像空间域 LSB 替换隐写和卡方隐写分析方法。

1.3 实验环境

操作系统：Windows 或 Linux
编程语言：Matlab

1.4 实验内容

（1） 采用 JPEG 格式图像作为载体图像，用 MATLAB 实现 JPEG 图片变换，使用 LSB 隐写方法，按顺序隐写的方式将秘密信息按位替换到载体中，比较图像信息隐写前后的直方图视觉效果。

（2） 通过计算图像像素值的卡方检验统计量，判断载体中是否存在隐藏信息。通过卡方检验量化隐写后存在的偏差，当 p 值低于阈值时判定为含密载体。

二、 实验一过程

2.1 系统结构设计

系统主要分为五个模块：**输入模块**（读取 JPEG 格式的载体图像和秘密信息文件，以二进制形式存储） -> **LSB 隐写模块**（将秘密信息的二进制数据嵌入载体图像像素的最低有效位） -> **图像保存模块**（将嵌入秘密信息的图像保存为新文件） -> **卡方分析模块**（计算隐写前后图像的直方图，基于卡方检验统计量分析是否存在隐藏信息） -> **输出模块**（输出隐写前后图像、局部直方图及卡方分析的 p 值）。

各模块之间相互连接传递作用，整个系统得以运行作用，接口函数、接口数据结构主要体现在：

- ✧ 输入模块与 LSB 隐写模块：输入模块通过 imread 函数读取 JPEG 图像，传递图像矩阵；秘密信息通过文件读取函数以二进制形式传递给 LSB 隐写模块。

- ✧ LSB 隐写模块与图像保存模块：LSB 隐写模块输出修改后的图像矩阵，图像保存模块通过 imwrite 函数保存为 JPEG 文件。
- ✧ 卡方分析模块与输出模块：卡方分析模块输出直方图数据和 p 值，输出模块通过 figure 和 plot 函数显示直方图及 p 值。

2.2 详细设计

（1）LSB 替换隐写算法

图像隐写技术能够将秘密信息嵌入图像载体而不显著影响其视觉质量，使第三方难以察觉信息存在。最低有效位（LSB）隐写是一种经典方法，通过修改像素灰度值的最低二进制位实现信息嵌入。由于最低有效位对像素值的影响极小（仅改变 2^0位权重），在灰度图像中（单通道 8bit 数据，取值范围 0-255），这种修改无法通过人眼直接察觉，因此适合作为秘密信息的载体。而 LSB 隐写算法主要有两种实现方式，下面我将分别进行更加详细地阐述。

1）顺序选取图像载体像素

该方法按图像像素的空间顺序依次嵌入秘密信息。首先读取灰度图像，获取像素灰度值矩阵。通过位运算（如与 254（二进制 11111110）按位与）将所有像素的最低位清零，为嵌入信息做准备。将秘密信息文件转换为单比特流（0/1 序列）。从第一个像素开始，将当前像素的最低位替换为比特流中的对应位。例如，若嵌入比特为 1，则将清零后的灰度值加 1；若为 0，则保持原值。嵌入完成后，通过 imhist 函数生成直方图，可视化嵌入前后像素灰度值的分布变化。其算法流程图如图 1-1 所示。

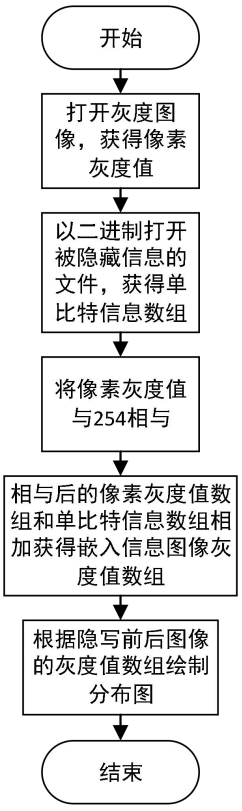

图 1-1 顺序选取图像载体像素的 LSB 算法流程图

此方法的特点是操作直接，但可能导致嵌入后的图像出现 “值对” 现象（即像素值为偶数和奇数的数量趋于均衡），原因在于秘密信息的 0/1 比特通常近似均匀分布，替换最低位后会使相邻灰度值（如 2i 和 2i+1）的像素数量接近。

2）随机选取图像载体像素

为减少 “值对” 现象的可检测性，随机嵌入法通过伪随机生成器选择嵌入位置。首先以固定种子（如 1）初始化伪随机生成器，生成一组随机距离值。通过累加距离值（cumsum）确定嵌入位置，确保位置不重复且覆盖全图（若总距离超过像素数，则截断最后一个距离）。然后创建与像素数量等长的掩码向量 flag，初始值为 255（二进制 11111111），将选中位置的掩码值设为 254（二进制 11111110），用于保留最低位以外的像素值。将原始像素灰度值与掩码向量按位与，清除选中位置的最低位，再逐位加上秘密信息的比特流，完成嵌入。随机化嵌入位置可分散修改痕迹，降低统计检测的敏感性。其算法流程图如图 1-2 所示。

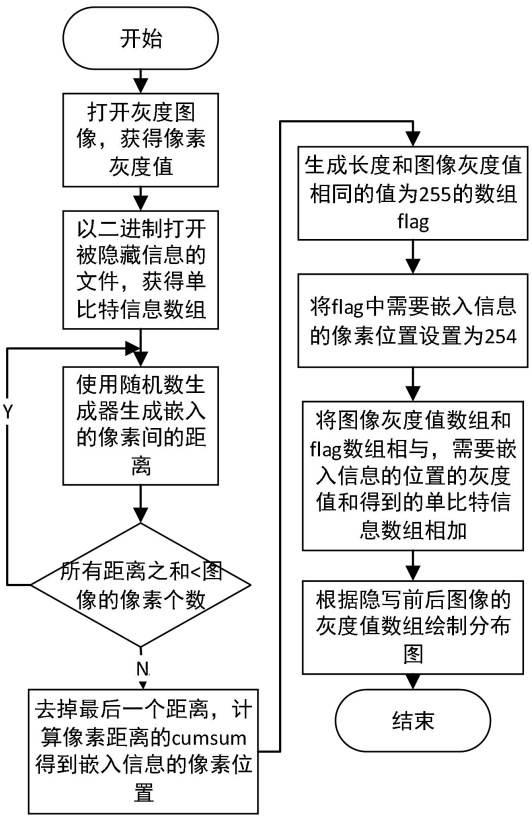

图 1-2 随机选取图像载体像素的 LSB 算法流程图

（2）卡方检验分析算法

上述两种方法均可以是生成嵌入信息的图像。若在未经过隐写的图像中像素值为 2i 和 2i+1 的像素点的数量相差很大，秘密信息视作为 0、1 随机分布的比特流，并且 0、1 比特概率各为 50%。若秘密信息完全替换了载体图像的最低位，则像素值为 2i 和 2i+1 的像素点数量会比较接近，产生“值对”现象，可以通过该性质判断图像是否经过隐写。顺序选取嵌入中若嵌入的信息流比较随机，则“值对”会更加明显。随机选取嵌入中的“值对”现象会大大减少。

可以通过卡方检验定量分析载体图像通过 LSB 顺序选取的方式嵌入秘密信息的情况。h_i 表示像素值为 i 的像素点总量。则卡方检验的计算过程为：

$$h_{2i}^{*}=\frac{h_{2i}+h_{2i+1}}{2}$$

$$q=\frac{h_{2i}-h_{2i+1}}{2}$$

$$r=\frac{\sum_{i=1}^{k}(h_{2i}-h_{2i}^{*})^{2}}{h_{2i}^{*}}$$

其中h_{2i}^{*}在隐写前后不会发生改变，r服从卡方分布，为卡方统计量，r越小表示嵌入信息的可能性越大。结合卡方分布密度函数计算被隐写的可能性为：

$$p=1-\frac{1}{2^{\frac{k-1}{2}}\Gamma(\frac{k-1}{2})}\int_{0}^{r}\exp(-\frac{t}{2})t^{\frac{k-1}{2}-1}dt$$

若 p 接近 1，则说明载体图像中含有秘密信息，该过程可以通过 MATLAB 中的统计分析库中的 chi2cdf 函数实现。

在进行编程时，只需要根据以上公式计算出其中的h_{2i}^{*}和r，然后通过 chi2cdf 函数计算得到 p 进行判断，若 p>0.8 则说明图像嵌入了信息，反之未嵌入。算法流程图如图 1-3 所示。

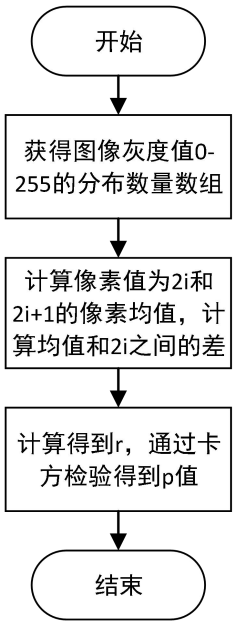

图 1-3 卡方隐写分析算法流程图

2.3 代码实现

该实验种我是通过 MATLAB 代码实现了基于最低有效位（LSB）的图像隐写术及卡方分析检测，下面是对核心代码的分析。首先是图像读取和预处理，读取灰度图像并显示其前 21 个灰度级的直方图，使用 imhist 获取直方图统计，stem 函数绘制离散直方图。然后是 LSB 信息嵌入模块，将二进制秘密信息嵌入图像像素的最低位，代码如下所示：

```
pos=1;

for col=1:width
    for row=1:length
        hidden_data1(col,row)=hidden_data1(col,row)-mod(hidden_data1(col,row),2)+msg(pos,1);
        if pos==len
            break;
        end
        pos=pos+1;
    end
    if pos==len
        break;
    end
end
```

逐像素遍历图像，通过模运算清除当前像素的最低位，再将秘密信息的对应比特写入最低位，最后嵌入完成后提前终止循环以节省计算资源。

接着把隐写图像保存并分析其最低位分布，其中 **bitand(hidden_data1,1)**提取最低位，放大 128 倍便于可视化和 **bitand(hidden_data1,128)**提取最高位进行对比。最后是实现卡方分析检测模块，对原始图像和隐写图像进行卡方分析，计算 P 值，并进行可视化输出，代码如下：

```
% origin
data = imhist(Picture_old)';
k=0;
r=0;
for i=1:256/2
    h_star = (data(2*i)+data(2*i-1))/2;
    if h_star~= 0
        k=k+1;
        r = r + (data(2*i)-h_star)*(data(2*i)-h_star)/h_star;
    end
end

P = 1-chi2cdf(r, k-1);
disp(P);
```

```
% new
data = imhist(Picture_new)';
k=0;
r=0;
for i=1:256/2
    h_star = (data(2*i)+data(2*i-1))/2;
    if h_star~= 0
        k=k+1;
        r = r + (data(2*i)-h_star)*(data(2*i)-h_star)/h_star;
    end
end

P = 1-chi2cdf(r, k-1);
disp(P);
```

三、 实验一测试与分析

3.1 系统测试及结果说明

（1）顺序选取图像载体像素实验结果及分析

在顺序选取像素的隐写实验中，隐写前后的图像视觉对比如图 1-4 所示。从直观视觉效果看，两幅图像无显著差异，表明 LSB 隐写技术通过修改最低位实现了信息嵌入的不可感知性，成功达成隐写的基本目标。

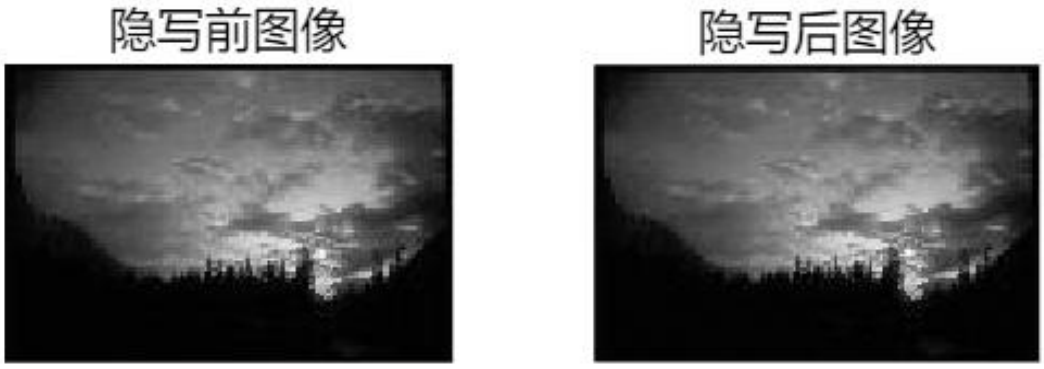

图 1-4 顺序选取像素隐写前后图像对比图

隐写前后的灰度值分布如图 1-5 所示。分析可知，原始图像中灰度值为偶数（如 2）和相邻奇数（如 3）的像素数量存在明显差异，而嵌入秘密信息后，奇偶灰度对（如 2/3、4/5 等）的分布趋于均衡，呈现典型的“值对”现象。这一现象的产生原因在于：秘密信息的 0/1 比特流通常近似均匀分布，当逐像素顺序替换最低位时，会强制将原始像素的奇偶分布调整为接近 50% 的均衡状态。例如，原始图像中灰度值为 2 的像素数量显著多于 3，嵌入信息后，部分 2 被修改为 3（或反之），使得两者数量接近。

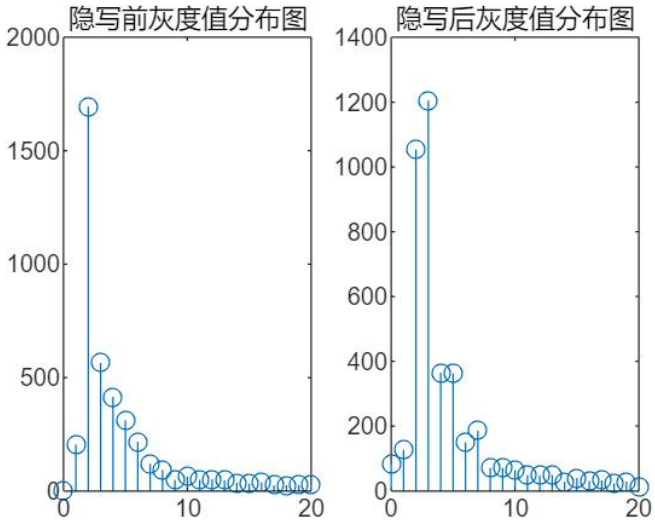

图 1-5 顺序选取像素隐写前后灰度值分布图

卡方检验结果如图 1-6 所示，计算得到的 p 值为 0.9948，非常接近 1。根据卡方检验的判定规则，p 值越接近 1，表明像素分布越偏离自然状态，隐写存在的可能性越高。该结果印证了顺序嵌入法会显著改变图像的统计特征，从而易于被基于统计规律的检测算法识别。

```
命令行窗口
>> test
p    =    0.9948
```

图 1-6 运行输出 p 值结果

（2）随机选取图像载体像素实验结果及分析

随机选取像素的隐写实验中，隐写前后的图像视觉效果如图 1-7 所示，肉眼同样无法辨别差异，说明随机嵌入法同样保持了良好的不可感知性。

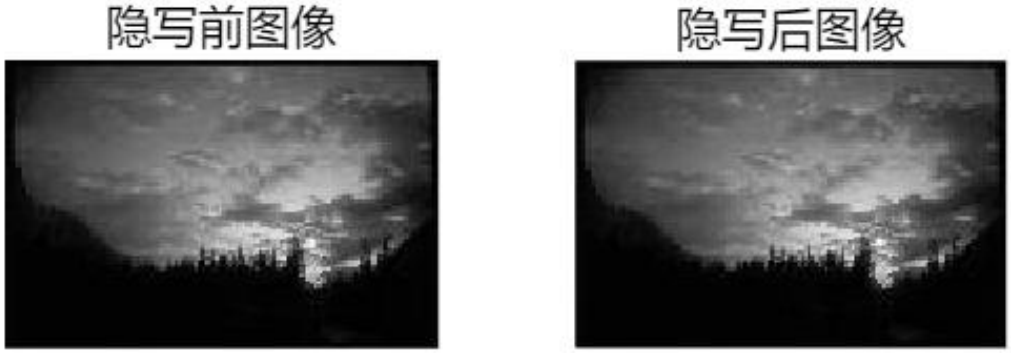

图 1-7 随机选取像素隐写前后图像对比图

对比灰度值分布（图 1-8）可见，隐写后图像的灰度直方图与原始图像相比虽有变化，但并未呈现明显的“值对”均衡现象。这一特性源于随机嵌入法的位置选择机制：

空间相关性：自然图像中相邻像素的灰度值通常具有强相关性（如平滑区域的像素值相近），而随机选取的像素位置可能跨越大范围空间，其灰度值差异较大，奇偶对的自然分布本就不均衡；

嵌入位置分散：随机化的嵌入位置避免了集中修改相邻像素，降低了奇偶对同步调整的概率，从而削弱了“值对”现象的统计显著性。

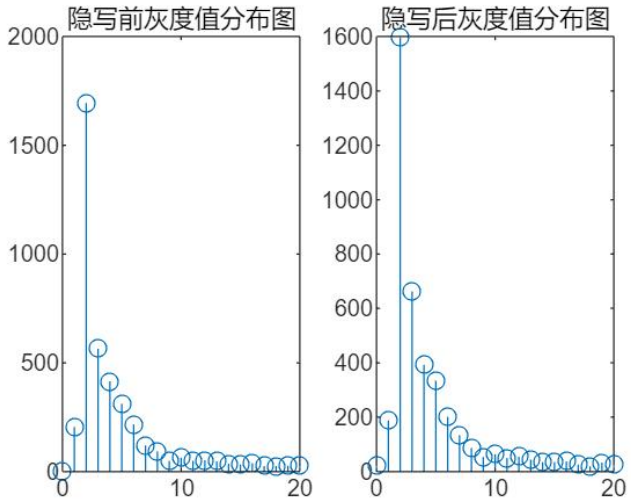

图 1-8 随机选取像素隐写前后灰度值分布图

卡方检验结果显示，该实验的 p 值趋近于 0，表明像素分布与自然状态的偏差极小，难以通过统计规律检测到隐写痕迹。这一结果验证了随机嵌入法通过分散修改位置，有效抑制了顺序嵌入中典型的“值对”效应，显著提升了隐写的抗检测能力。

3.2 遇到的问题及解决方法

问题 1：秘密信息长度超过图像像素容量，导致嵌入失败。

解决方法：在 LSB 隐写函数中添加容量检查，确保秘密信息长度不超过图像像素数。

问题 2：卡方分析 p 值计算不准确，部分图像未检测出隐写信息。

解决方法：检查直方图计算逻辑，确保期望频数不为零时才累加卡方统计量，避免除零错误。

3.3 设计方案存在的不足

- LSB 隐写方法对图像压缩敏感，可能在高压缩率 JPEG 图像上失效。
- 卡方分析仅适用于顺序嵌入的 LSB 隐写，对随机嵌入效果较差。
- 未考虑彩色图像的 RGB 通道处理，限制了应用场景。

实验二 图像 LSB 隐写与分析方法

四、 实验二概述

4.1 实验名称

图像 JSTEG、F3、F4 隐写方法

4.2 实验目的

掌握图像 DCT 变换域基本隐写方法。

4.3 实验环境

操作系统：Windows 或 Linux
编程语言：Matlab

4.4 实验内容

（1） JSTEG 隐写方法，选择非零且不等于 ±1 的量化后 DCT 系数，并替换其最低有效位（LSB）以嵌入秘密数据。

（2） F3 隐写方法，操作量化后的 DCT 系数来嵌入信息：若系数的 LSB 与待嵌入的比特不匹配，则将该系数向零方向递减（不直接翻转 LSB），若递减后变为零则视为无效并跳过该比特。

（3） F4 隐写方法，操作量化后的 DCT 系数来嵌入信息：当 DCT 系数为正值，若奇偶性不匹配则绝对值减 1，否则不变；当 DCT 系数为负值，若奇偶性匹配则绝对值减 1，否则不变，若递减后变为零则视为无效并跳过该比特。

五、 实验二过程

5.1 系统结构设计

系统主要分为以下六个模块：**输入模块**（读取 JPEG 图像和秘密信息文件，以二进制形式存储） -> **DCT 解码模块**（对 JPEG 图像进行部分解码，提取量化后的 DCT 系数） -> **隐写模块** JSTEG/F3/F4（根据 JSTEG、F3 或 F4 算法规则，将秘密信息嵌入 DCT 系数） -> **图像编码模块**（对修改后的 DCT 系数进行编码，生成隐写图像） -> **卡方分析模块**（计算隐写前后 DCT 系数直方图，基于卡方检验分析隐写效果） -> **输出模块**（输出隐写前后图像、DCT 系数直方图及卡方分析结果）。

该实验与上一个实验类似，重复部分便不再赘述：

- ✧ 输入模块与 DCT 解码模块：输入模块通过 imread 读取 JPEG 图像，传递给 DCT 解码模块提取 DCT 系数矩阵；秘密信息以二进制流形式传递。
- ✧ DCT 解码模块与隐写模块：DCT 解码模块输出量化后的 DCT 系数矩阵，隐写模块根据算法规则修改系数。
- ✧ 隐写模块与图像编码模块：隐写模块输出修改后的 DCT 系数矩阵，图像编码模块通过 imwrite 生成 JPEG 图像。

我们可以通过表格的形式来对三个隐写算法进行更加直观的对比分析，如下表所示：

表 2-1 JSTEG/F3/F4 隐写算法对比分析

算法	嵌入方式	特点	抗检测能力	提取复杂度
JSTEG	直接修改 LSB	简单高效	弱	低
F3	调整系数绝对值	避免生成零系数	中	低
F4	区分正负系数修改	更好的统计隐蔽性	强	中

5.2 详细设计

（1）JSTEG 图像隐写和提取

JSTEG 隐写方案不同于基于像素灰度值的 LSB 方法，而是在图像离散余弦变换（DCT）域内实现信息嵌入。对于 JPEG 格式图像，其压缩过程包含色彩空间转换（RGB→YUV）与 DCT 变换两个核心步骤。本实验采用灰度图像，处理流程聚焦于明度信息：首先将像素灰度值归一化（减去 128），随后划分成 8×8 子块，通过 DCT 变换获取各子块的频率系数矩阵。其中，矩阵左上角的直流（DC）系数表征低频信息，数值较大；其余交流（AC）系数代表高频细节，绝对值较小且多为零值，对视觉影响微弱。量化与哈夫曼编码等后续压缩操作，进一步降低数据冗余度。JSTEG 算法选择 AC 系数作为载体，具体嵌入步骤如下：

1) DCT 域转换：对 JPEG 文件执行 DCT 变换，获取 8×8 系数矩阵；
2) 区域筛选：排除 DC 系数及特定行列（如第 1 行、第 1 列及间隔 8 行的区域），仅对剩余 AC 系数操作；
3) 嵌入规则：遍历 AC 系数，跳过数值为 0 或 1 的元素，仅对其余系数的最低有效位进行替换，嵌入秘密信息；
4) 格式转换：将修改后的 DCT 矩阵重新编码，保存为 JPEG 格式。

其具体的算法流程图如图 2-1 所示。

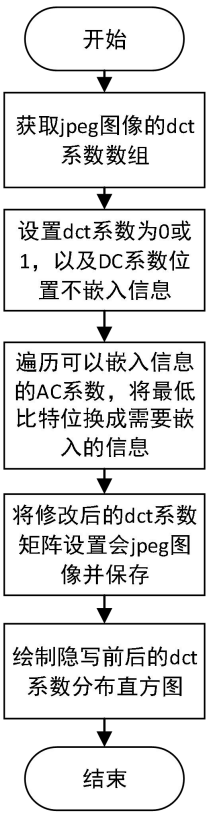

图 2-1 JSTEG 图像隐写算法

JSTEG 图像隐写的逆过程及提取算法首先获得 jpeg 图像的 dct 矩阵，设置 dct 系数为 0 和 1 的位置，和 DC 系数位置为未嵌入信息的位置。顺序遍历嵌入信息位的 AC 系数，取其最低比特位，将最终遍历得到的所有最低比特位组成最终隐写的信息，其算法流程图如图 2-2 所示。

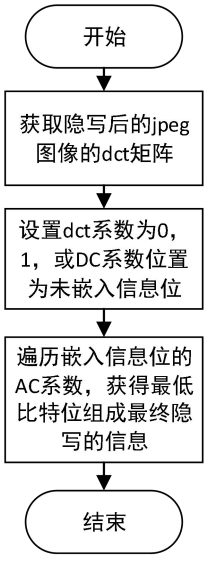

图 2-2 JSTEG 图像隐写的提取算法

（2）F3 图像隐写和提取

F3 算法与 JSTEG 均基于 DCT 域，但在嵌入规则与无效值处理上存在显著差异：

- 嵌入规则优化：仅排除 AC 系数为 0 的元素，允许对 ±1 值进行操作；
- 奇偶性嵌入：根据 AC 系数正负与秘密比特的关系调整数值：

- ◆ 当 AC > 0 时，若 AC 奇偶性与嵌入比特一致则不变，否则减 1；
- ◆ 当 AC < 0 时，若 AC 奇偶性与嵌入比特一致则不变，否则加 1；

■ 无效操作处理：若修改后 AC 系数变为 0（如 ±1 值与嵌入比特冲突），则跳过该位置，继续下一个系数的嵌入。

具体的隐写算法流程图如图 2-3 所示。

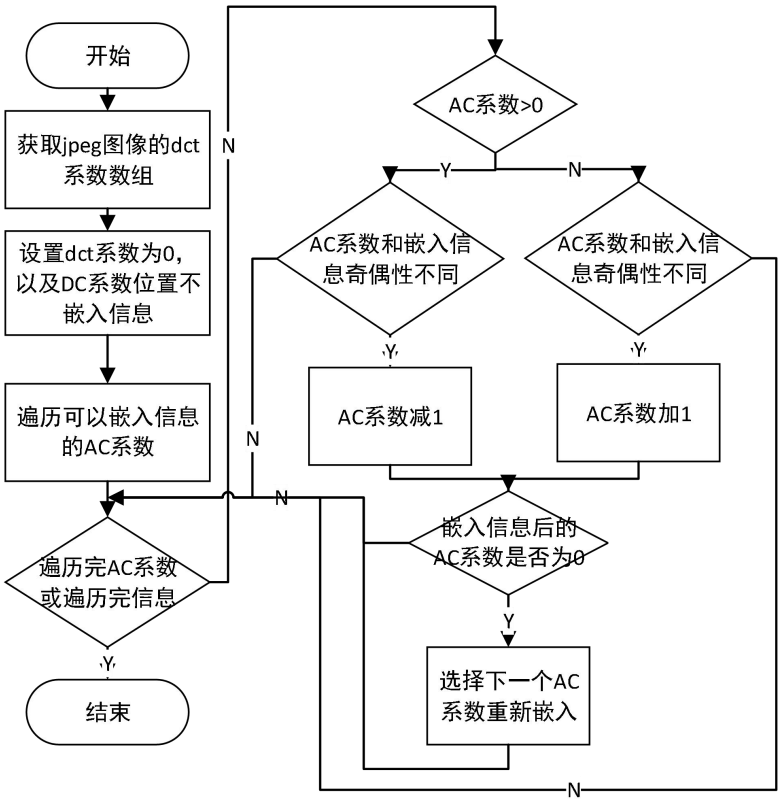

图 2-3 F3 图像隐写算法流程图

F3 隐写算法的利用奇偶性进行隐写的巧妙之处在于可以通过隐写后的 AC 系数的最低比特位直接还原出原始信息，不用进行转换，所以其算法流程图如图 2-4 所示。

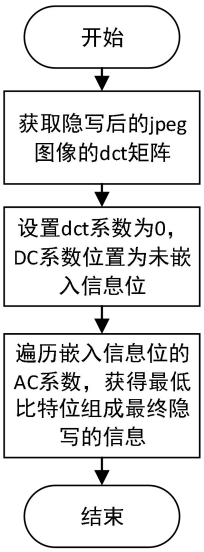

图 2-4 F3 图像隐写的提取算法流程图

（3）F4 图像隐写和提取

F4 算法在 F3 的基础上进一步改进，核心差异体现在负数 AC 系数的处理逻辑：

- 嵌入规则细化：当 AC < 0 时，若其奇偶性与嵌入比特不一致则加 1，否则保持不变；同时，若原始系数为 ±1 且与嵌入比特冲突导致修改后为 0，则判定本次嵌入无效，跳转至下一位置；
- 提取逻辑扩展：提取时需先判断 AC 系数的正负性：
  - 若 AC > 0，直接读取最低位作为秘密比特；
  - 若 AC < 0，对最低位执行异或操作（与 1 异或）后获取秘密比特。

具体的算法流程图如图 2-5 和 2-6 所示。

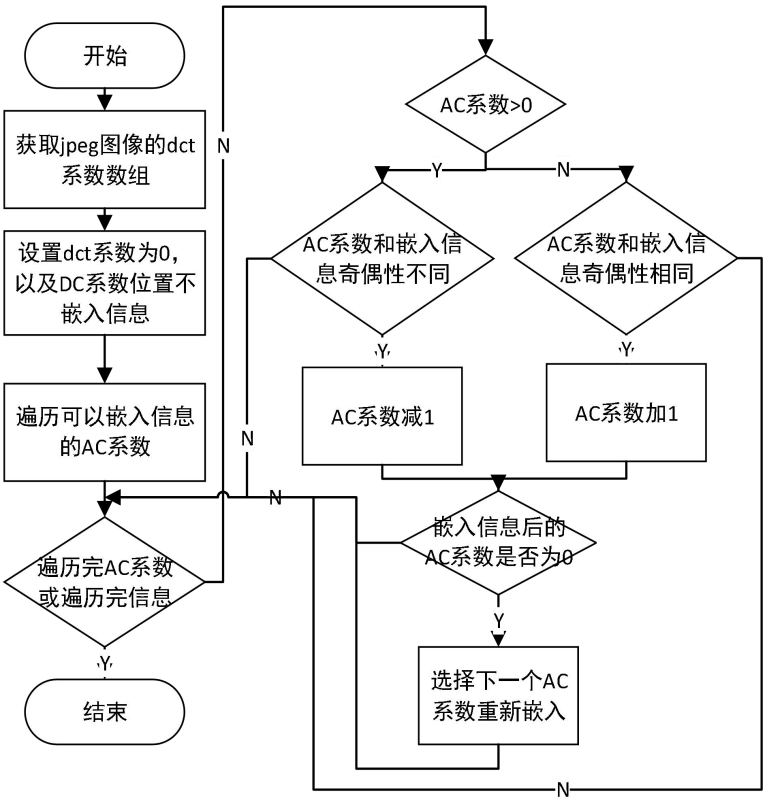

图 2-5 F4 图像隐写算法流程图

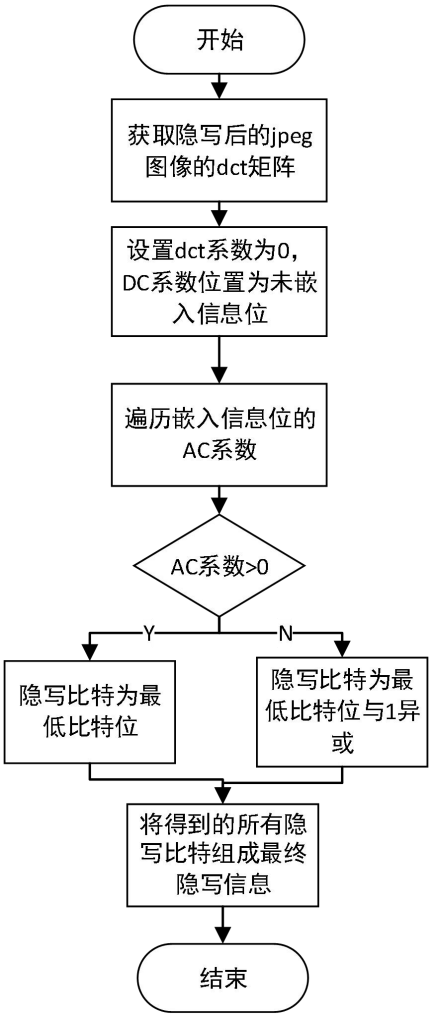

图 2-6 F4 图像隐写的提取算法

相较于 F3，F4 通过增加正负性判断机制，在保持高嵌入容量的同时，进一步优化了统计隐蔽性，但提取过程复杂度有所提升。

5.3 代码实现

该实验我同样是使用 MATLAB 代码实现了三种 JPEG 图像隐写算法（JSTEG、F3、F4）及其对应的卡方分析检测，下面是对核心代码的详细解释：

JSTEG 隐写算法在 JPEG 图像的 DCT 系数中，选择绝对值大于 1 的 AC 系数，修改其最低有效位嵌入信息。

嵌入函数：修改 AC 系数的 LSB

```
% 定位可嵌入的AC系数（非DC、绝对值>1）
changeable = true(size(DCT));
changeable(1, 1) = false; % DC系数不嵌入
changeable(abs(DCT) <= 1) = false; % 跳过绝对值≤1的系数
posAC = find(changeable);
ncAC = numel(posAC); % 可嵌入位置总数

messageLen = length(message);
if messageLen > ncAC
    error('ERROR: 信息长度超过载体容量');
end

% 逐位嵌入信息
for i_MSG = 1:messageLen
    idx = posAC(i_MSG);
    coeff = DCT(idx);
    % 修改最低有效位
    if coeff > 0
        new_coeff = coeff - mod(coeff, 2) + message(i_MSG);
    else
        new_coeff = coeff - mod(coeff, 2) - message(i_MSG);
    end
    DCT(idx) = new_coeff;
end
```

提取函数：读取 LSB

```
    message(i_MSG) = mod(abs(coeff), 2); % 取绝对值的最低位
```

F3 隐写算法区分正负系数，负数系数通过异或操作嵌入，进一步降低统计特征变化，**相比 JSTEG，减少了对 DCT 系数的扰动**。

嵌入函数：基于奇偶性的 ±1 调整

```
% 定位非零AC系数
changeable = true(size(DCT));
changeable(1, 1) = false;
changeable(DCT == 0) = false; % 排除0，允许±1
posAC = find(changeable);

% 逐位调整系数奇偶性
for i_MSG = 1:messageLen
    idx = posAC(i_MSG);
    coeff = DCT(idx);
    curr_lsb = mod(abs(coeff), 2);
    if curr_lsb ~= message(i_MSG)
        new_coeff = coeff - sign(coeff); % 奇偶性不符时减/加1
        if new_coeff ~= 0 % 避免生成0，否则回退
            DCT(idx) = new_coeff;
        else
            i_MSG = i_MSG - 1; % 回退重嵌
        end
    end
end
```

提取函数：直接读取奇偶性

```
% 读取非零AC系数的绝对值LSB
message = mod(abs(DCT(posAC(1:messageLen))), 2);
```

F4 隐写算法区分正负数进行嵌入，进一步减少统计特征的改变，正数为修改 LSB 与消息位一致，负数则为相反，**相比 F3，具有更好的统计隐蔽性**。

嵌入函数：正负系数差异化调整

```
% 定位非零AC系数
changeable = true(size(DCT));
changeable(1, 1) = false;
changeable(DCT == 0) = false; % 排除0，允许±1
posAC = find(changeable);

% 逐位调整系数奇偶性
for i_MSG = 1:messageLen
    idx = posAC(i_MSG);
    coeff = DCT(idx);
    curr_lsb = mod(abs(coeff), 2);
    if curr_lsb ~= message(i_MSG)
        new_coeff = coeff - sign(coeff); % 奇偶性不符时减/加1
        if new_coeff ~= 0 % 避免生成0，否则回退
            DCT(idx) = new_coeff;
        else
            i_MSG = i_MSG - 1; % 回退重嵌
        end
    end
end
```

提取函数：负数 LSB 取反

```
% 正数直接取LSB，负数取LSB异或1
for i_MSG = 1:messageLen
    coeff = DCT(posAC(i_MSG));
    if coeff > 0
        message(i_MSG) = mod(coeff, 2);
    else
        message(i_MSG) = mod(abs(coeff), 2) ^ 1; % 负数LSB取反
    end
end
```

六、 实验二测试与分析

6.1 系统测试及结果说明

（1）JSTEG 图像隐写和提取

JSTEG 算法隐写前后的图像对比如图 2-7 所示，视觉上无显著差异，表明该算法通过修改 DCT 域高频系数实现了信息嵌入的不可感知性。

图 2-7 JSTEG 隐写前后图像对比

隐写前后的 DCT 系数分布直方图（图 2-8）显示出两个特征：

• **0 和 1 值的稳定性**：由于算法跳过数值为 0 或 1 的 AC 系数，这两类值的数量在隐写前后保持不变；

• **奇偶对均衡化**：非零非 1 的 AC 系数（如 ±2、±3 等）的奇偶对（如 2/3、-2/-3）数量显著趋近，呈现 “成对效应”。这是因为秘密信息的 0/1 比特随机分布，嵌入时通过修改最低位强制均衡奇偶分布，与实验一中 LSB 随机嵌入的统计特性相似。

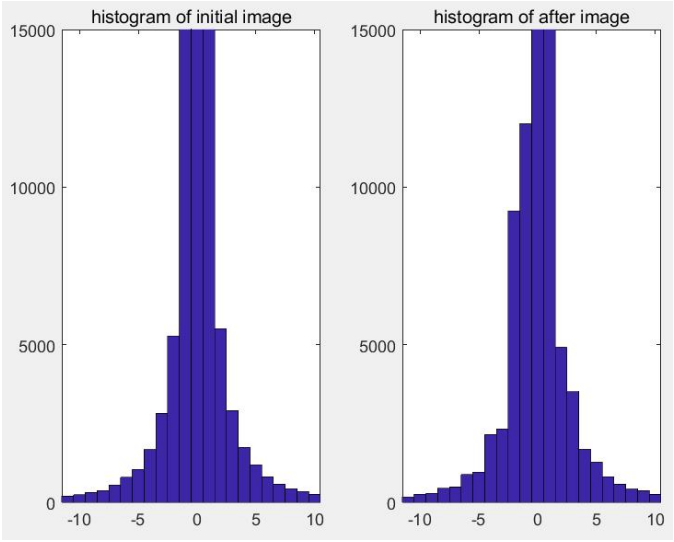

图 2-8 JSTEG 隐写前后 DCT 系数分布直方图

JSTEG 的嵌入操作主要影响非零非 1 的 AC 系数，虽未改变 0/1 值的分布，但通过奇偶对均衡化引入了可检测的统计异常。

（2）F3 图像隐写和提取

F3 算法隐写前后的图像视觉效果如图 2-9 所示，依然保持高度一致性。

图 2-9 F3 图像隐写前后的图像对比

其 DCT 系数直方图（图 2-10）呈现以下特征：

• **零值数量增加**：由于算法允许对 ±1 值进行嵌入操作，当嵌入比特与 ±1 的奇偶性冲

突时（如 1 嵌入 0 需减 1 变为 0），会导致零值数量显著上升；

• **±1 值减半**：原始 ±1 值约有一半因嵌入操作被修改为 0 或 ±2，故其数量近似减少 50%；

• **偶数值偏多现象**：直方图中偶数值（2i）的柱体高度高于相邻奇数值（2i-1）。这是由于无效嵌入机制的存在 —— 当 AC 系数为 ±1 且需嵌入 0 时，算法会跳过该位置并尝试下一系数，导致后续嵌入位置更倾向于偶数值，使得 0 的嵌入比例高于 1（假设其他系数嵌入 0 的概率约为 60%），最终形成偶数值主导的分布特征。

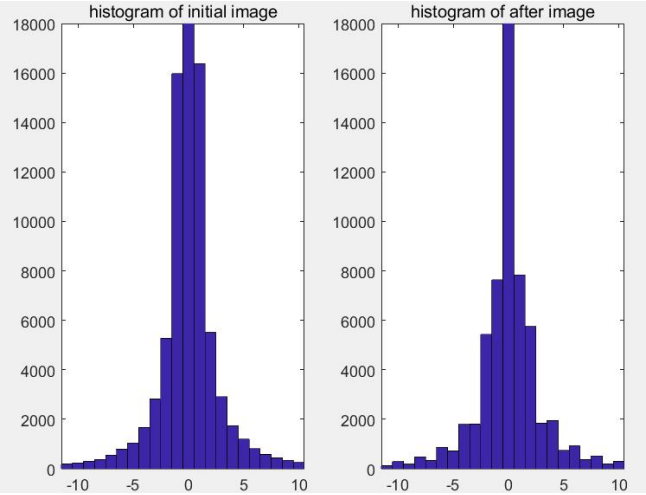

图 2-10 F3 隐写前后的 DCT 系数分布直方图

F3 通过动态调整系数而非直接修改最低位，一定程度上降低了统计异常的显著性，但无效嵌入机制引入了新的分布偏差。

（3）F4 图像隐写和提取

F4 算法隐写前后的图像对比（图 2-11）同样无法通过肉眼区分，体现了其良好的不可见性。

图 2-11 F4 隐写前后图像对比

DCT 系数直方图（图 2-12）显示：

• **奇偶分布均衡**：与 F3 不同，F4 中偶数值（2i）与奇数值（2i-1）的柱体高度基本持平，

未出现显著偏向。这是因为 F4 对负数系数采用“异或式”嵌入规则（负数的最低位需取反），使得 0 和 1 的嵌入概率在正负系数中趋于均衡，避免了 F3 中因无效嵌入导致的单向偏差；

• **零值与 ±1 值的稳定性**：尽管 F4 也存在对 ±1 值的处理，但通过引入无效嵌入判定（如 ±1 修改后为 0 则跳过），减少了零值的生成，使得 ±1 值的数量变化较 F3 更平缓。

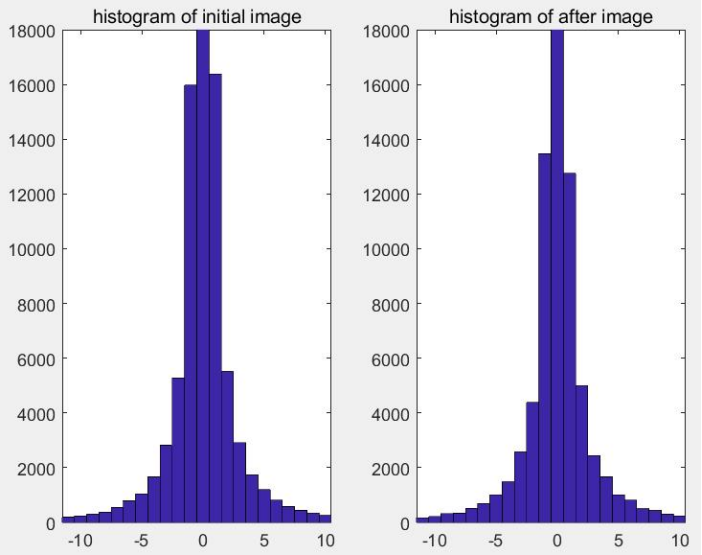

图 2-12 F4 隐写前后图像的 DCT 系数分布直方图

F4 通过正负系数差异化的嵌入逻辑，有效抑制了统计特征的异常波动，其直方图分布更接近自然状态，抗检测能力显著优于 JSTEG 和 F3。

6.2 遇到的问题及解决方法

问题 1：DCT 系数提取不准确，导致隐写失败。

解决方法：使用 Matlab 的 dct2 函数正确提取 DCT 系数，确保量化步骤一致。

问题 2：F3/F4 隐写中部分系数变为零，影响嵌入容量。

解决方法：在算法中跳过无效嵌入（系数变为零），并检查秘密信息长度是否超出容量。

问题 3：卡方分析对 F4 隐写的检测效果较差。

解决方法：优化直方图计算范围，聚焦 DCT 系数的主要分布区间（-50 到 50）。

6.3 设计方案存在的不足

- JSTEG：成对效应明显，易被卡方分析检测，隐蔽性较低。
- F3：偶系数增多现象仍可被统计分析检测，隐蔽性有待提升。
- F4：虽然隐蔽性最佳，但嵌入容量较低，计算复杂度较高。
- 实验未涉及彩色图像的 RGB 通道处理，限制了算法的通用性。

七、 实验总结

7.1 实验感想

实验一：通过本次实验，我明白了图像隐写的一个基本过程，学会了 LSB 替换隐写的原理与实现方法，也对 MATLAB 的一些基本操作有了一定的掌握。在进行卡方隐写分析时，我对隐写过程有了更加深入的了解。秘密信息并不是被隐藏起来就一定安全，通过使用某些特殊的方法仍然可以回溯出被隐藏的信息。这也让我对信息安全的重要性有了进一步认识。

实验二：通过本次实验，我深入掌握了 JSTEG、F3 和 F4 隐写算法的原理与实现，理解了 DCT 变换域隐写的特点及其在隐蔽性与容量之间的权衡。JSTEG 简单高效但易检测，F3 改进了成对效应但引入新痕迹，F4 在安全性上表现最佳但牺牲了嵌入效率。这让我认识到隐写技术需要在隐蔽性、容量和计算复杂度间寻求平衡。

从思政角度看，隐写技术在保护信息安全的同时，也可能被用于非法用途，提醒我们在技术开发中需关注伦理责任，服务于国家和社会的安全需求。

7.2 意见和建议

原创性声明

本人郑重声明本报告内容，是由作者本人独立完成的。有关观点、方法、数据和文献等的引用已在文中指出。除文中已注明引用的内容外，本报告不包含任何其他个人或集体已经公开发表的作品成果，不存在剽窃、抄袭行为。

已阅读并同意以下内容。

判定为不合格的一些情形：

（1） 请人代做或冒名顶替者；

（2） 替人做且不听劝告者；

（3） 实验报告内容抄袭或雷同者；

（4） 实验报告内容与实际实验内容不一致者；

（5） 实验代码抄袭者。

作者签名：