数字水印实验二 图像LSB数字水印

实验原理

LSB隐写可以被看做是一种信息隐藏技术，通过改变数字媒体文件的像素中的最低有效位，将需要隐藏的信息嵌入到其中，从而实现隐蔽信息的目的。

在本实验中，由于载体图像与隐秘图像的大小相同，故我们采用的隐藏方式是最基础的连续像素点比特替换算法，即分别在8个“撕开”的位平面中，将所有像素点值用二值隐秘图像的对应像素点值替换（当然这8个位平面只有最低位的位平面才能用来进行真正意义上的lsb隐写）。

Code

Encode

encode.m Plain Text 复制代码

```
clc %清屏
clear %清空变量
close all %关闭已打开图像

plain_image = imread('Lenna.bmp');                %得到图像
secret_image = imread('woman.bmp');
plain_image = rgb2gray(plain_image);              %若图像是彩色，先要转化为灰度图像
image1=plain_image;
[height,width,s]=size(image1);                          %获取图像大小
subplot(3,3,1)
imshow(plain_image);
for n=1:8
    for i=1:height
        for j=1:width
            % a=bitget(plain_image(i,j),n);             %提取这个位的值
            b=bitget(secret_image(i,j),n);
            image1(i,j)=bitset(image1(i,j),n,b);
        end
    end
    sub = int2str(n);
    path = ['encode_result/',sub,'.bmp'];
    imwrite(image1,path);
    subplot(3,3,n+1)                            %循环显示
    imshow(image1)
    image1=plain_image;
end
```

首先将载体和隐秘图片读入，将彩色图片转化为灰度图像（相当于是一个降维操作）。

每个像素点的值是0-255，可以用8位二进制来编码，因此我们可以将整个图片“撕开”成8个位平面。最外层循环用来确定当前操作的位平面。对于每个位平面来说，我们遍历其宽度及高度，实现对于每个像素点的遍历。在当前像素点，我们不用管载体图像的像素值是多少，直接将其替换为隐秘图像的像素值。p.s.用matlab写隐写确实爽，bitget函数和bitset函数真好用。

将隐藏后的8个位平面图片依次打印并保存为“n.bmp”，其中n为替换位的least-significant程度。

Decode

decode.m Plain Text 复制代码

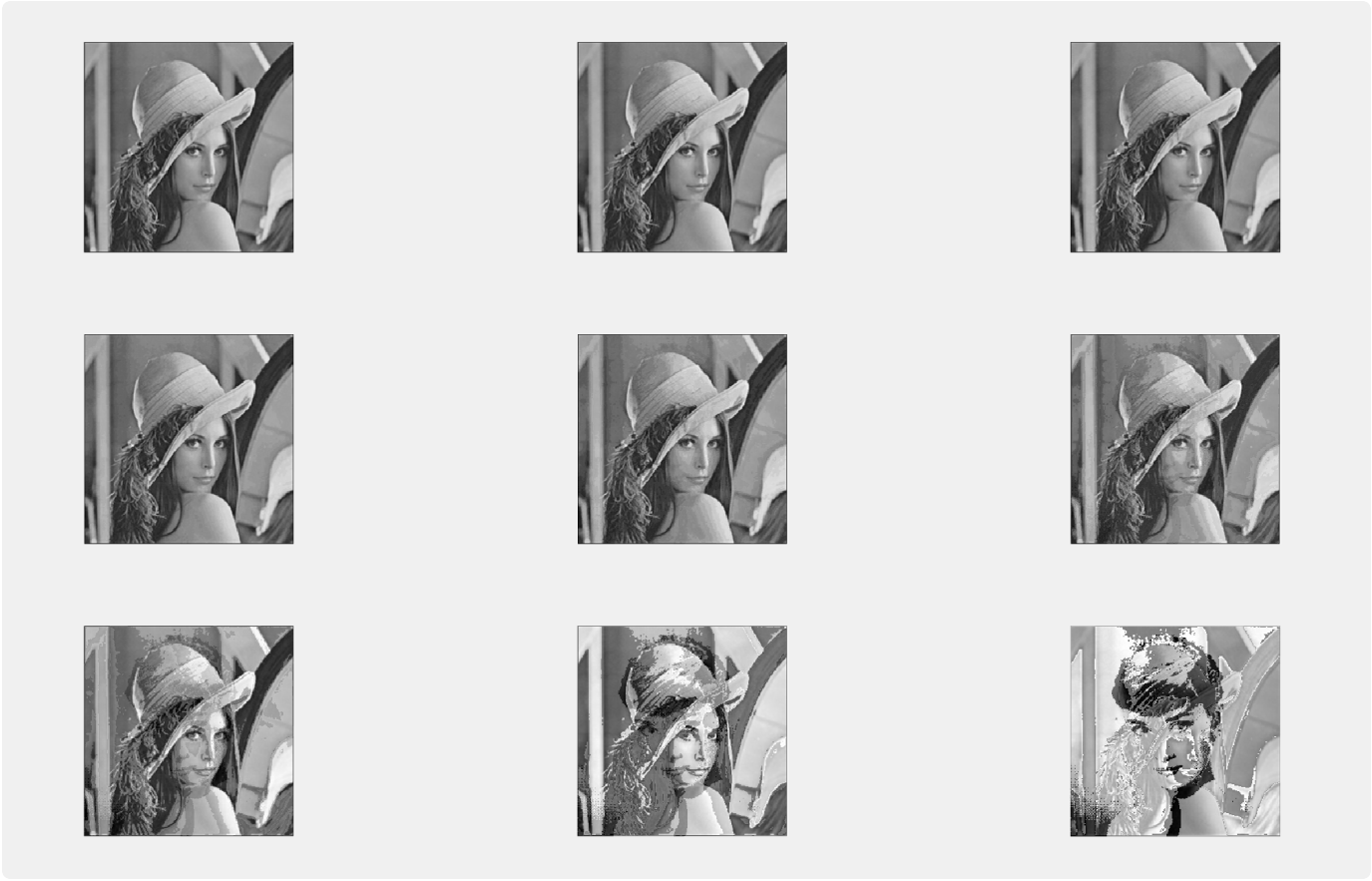
```
clc %清屏
clear %清空变量
close all %关闭已打开图像

plain_image = imread('Lenna.bmp');
lsb_image = imread('encode_result\1.bmp');
plain_image = rgb2gray(plain_image);

[height,width,s]=size(plain_image);
secret_image = zeros(height,width,1);
for i=1:height
    for j=1:width
        a=bitget(plain_image(i,j),1);              %提取这个位的值
        b=bitget(lsb_image(i,j),1);
        secret_image(i,j)=bitset(secret_image(i,j),1,b);
    end
end

imshow(secret_image);
imwrite(secret_image,'secret_image.bmp');
```

提取隐秘图片的过程也很简单。我们可以在8个含密图片中任意选取一个，这里我们选择了1.bmp，也就是真正实现了lsb隐写的那张含密图像。

提取的具体操作：我们准备一张与含密图像同样大小的空白图像 secret_image = zeros(height,width,1); 将含密图像每个像素点的最低有效位提取出来，转存到空白图像的对应像素点中，即可得到一张完整的二值图像，也就是提取出的秘密图像。

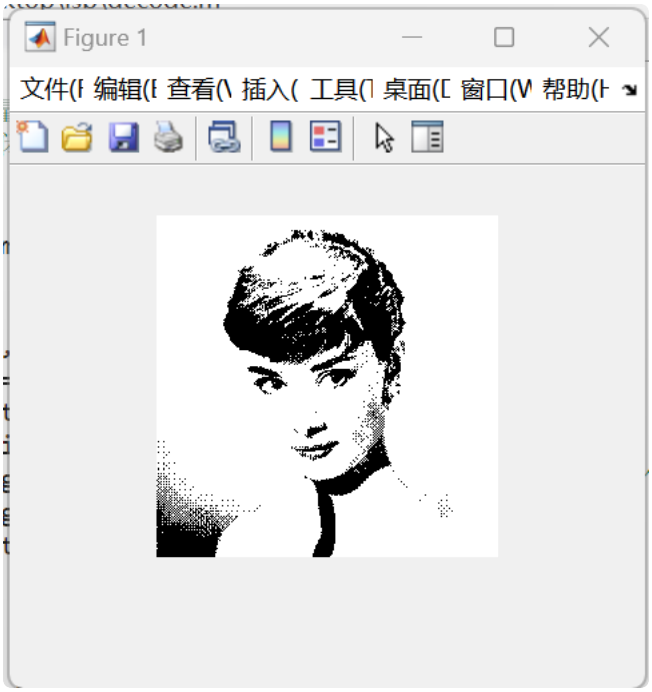
实验结果

隐藏结果



我们将隐藏结果用九宫格的形式展示，其中左上角的是未隐藏信息的原始图像。剩余的8张含密图像逐渐从lsb变成msb。可以看到，如果将信息插入到低4位的位平面中时，对于人眼来说的不可见行较强，高3位的隐藏则对图片本身的改变较显著。

提取结果



提取出的二值图片与加密时的秘密图像一致，提取成功。

实验分析&思考

如何提高鲁棒性

1. 块选择方法：采用选择块的方法而不是选择像素来嵌入秘密信息。
2. 多层嵌入方法：本实验在一定程度上提示了我们可以利用视觉的误差，在一定不可见行范围内采用多层嵌入方法将秘密信息分配到多个频率或分辨率的图像中。
3. 自适应调整方法：使用自适应调整方法可以选择嵌入数据的位置，并根据图像的局部特征动态调整嵌入算法参数。在本实验的woman.bmp中，人物的头发处全黑像素点较密集，可以考虑将其以某种可逆算法分散一下
4. 加密方法：在嵌入秘密信息时，使用一种加密方法可以确保只有授权人员才能访问信息。这种方法可以提高秘密信息的安全性，并且可防止未经授权的人员攻击和篡改信息。
5. 污点遮蔽方法：将秘密信息嵌入到图像中的特定区域，同时使用污点遮蔽方法隐藏嵌入点的位置，使得攻击者很难找到嵌入点。这种方法可以提高安全性和鲁棒性。

在https://github.com/abyssal-whale/lsb_steganography/tree/main/LSB-Steganograpy_python中，有本人使用python实现的较高鲁棒性隐写，利用了RSA加密和非连续替换。