

目 录

1	基于 LSB 的空域信息隐藏实现	3
1.1	问题描述.....	3
1.2	系统设计.....	3
1.3	系统实现.....	3
1.3.1	读取图片	3
1.3.2	嵌入	4
1.3.3	输出	4
1.3.4	提取	5
1.4	实验小结.....	6
2	JPEG 图像变换域信息隐藏实现	7
2.1	问题描述.....	7
2.2	系统设计.....	7
2.3	系统实现.....	8
2.3.1	行列解算	8
2.3.2	Jsteg 算法	8
2.3.3	F3 算法	9
2.3.4	F4 算法	10
2.3.5	F5 算法	11
2.3.6	运行结果	11
2.4	实验小结.....	14
3	指导教师评定意见	15
4	附录 A 基于 LSB 的空域信息隐藏实现的源程序.....	16
4.1	嵌入.....	16
4.2	提取.....	17
5	附录 B JPEG 图像变换域信息隐藏实现的源程序	18
5.1	Jsteg 嵌入.....	18

5.2	Jsteg 提取	20
5.3	F3 嵌入	21
5.4	F3 提取	24
5.5	F4 嵌入	25
5.6	F4 提取	29
5.7	F5 嵌入	30
5.8	F5 提取	35

1 基于 LSB 的空域信息隐藏实现

1.1 问题描述

LSB 空域信息隐藏算法关键步骤是将原始图像最低一个位平面替换为要隐藏的秘密信息。

1、要求实现信息嵌入算法。自选载体图像，嵌入内容为自己的学号。首先对原始图像中每个像素点的灰度值进行变换，由十进制转换为二进制；然后将秘密信息转换为二进制序列，并将原始图像中像素点的最低比特为替换为二进制序列中的每一比特信息；在上述替换结束后，将像素点的二进制数据转换回十进制数据，保存为含有秘密信息的图像，并与原始图像同框显示以对比视觉效果。

2、要求实现信息提取算法。从含有秘密信息的 BMP 图像中提取出自己的学号。首先对载密图像中每个像素点的灰度值进行变换，由十进制转换为二进制；然后提取图像像素点中最低有效位的数据，根据嵌入顺序进行组合得到嵌入秘密信息。

1.2 系统设计

嵌入基于 .bmp 格式图片。

参考设计样例，以行顺序进行嵌入明文的二进制字节流。

1.3 系统实现

1.3.1 读取图片

使用 `imread` 函数读取图片，并获取相关信息。

```
1.      Picture=imread('kinkakuji01.bmp');
```

```
2. Double_Picture=Picture;
3. Double_Picture=double(Double_Picture);
4. [m,n]=size(Double_Picture);
```

1.3.2 嵌入

以行顺序逐个嵌入最低位. 当嵌入完毕或图片被嵌入满时跳出.

```
1.         for f2=1:n
2.     for f1=1:m
3.         fprintf("[LOG] Embedding %d %d\t%d\n",f1,f2,Double_Picture(f1,f2));
4.         Double_Picture(f1,f2)=Double_Picture(f1,f2)-
            mod(Double_Picture(f1,f2),2)+msg(p);
5.         if p==len
6.             break;
7.         end
8.         p=p+1;
9.     end
10.    if p==len
11.        break;
12.    end
13. end
```

1.3.3 输出

整理数据, 输出到新的嵌入图片中. 并显示.

```
1. Double_Picture=uint8(Double_Picture);
2. imwrite(Double_Picture,'kinkakuji01-steg.bmp');
3. subplot(121);imshow(Picture);title('Original Pic');
4. subplot(122);imshow(Double_Picture);title('Embedded Pic');
```

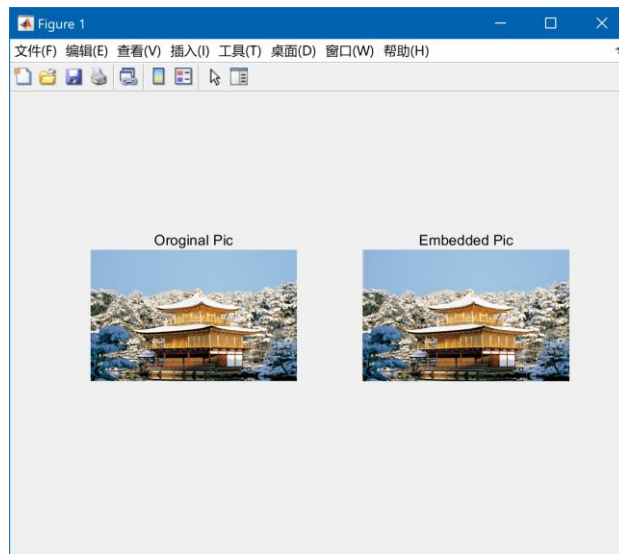


图 1-1 嵌入后的图片对比

1.3.4 提取

行顺序提取, 并将结果添加到结果字符串中.

```

1.  for f2=1:n
2.      for f1=1:m
3.          if bitand(Picture(f1,f2),1)==1
4.              result=result+"1";
5.          else
6.              result=result+"0";
7.          end
8.          if p==len
9.              break;
10.         end
11.         if p<len
12.             p=p+1;
13.         end
14.     end
15.     if p==len
16.         break;
17.     end
18. end
    
```

```
>> extract  
[SUCCESS] Msg=01000001011000110111001001101111011100110111001100100000011101000110100001100101001000000110011101
```

图 1-2 运行结果,和明文字节流吻合.

明文为 Across the Great Wall We can Reach Every Corner of the World U201714886

1.4 实验小结

LSB 算法是对空域的 LSB 做替换,用来替换 LSB 的序列就是需要加入的水印信息、水印的数字摘要或者由水印生成的伪随机序列。由于水印信息嵌入的位置是 LSB,为了满足水印的不可见性,允许嵌入的水印强度不可能太高。然而针对空域的各种处理,如游程编码前的预处理,会对不显著分量进行一定的压缩,所以 LSB 算法对这些操作很敏感。因此 LSB 算法最初是用于脆弱性水印的。

2 JPEG 图像变换域信息隐藏实现

2.1 问题描述

通过实验达到：

- (1) 加深对变换域信息隐藏算法原理的理解；
- (2) 熟悉数字图片 JPEG 压缩格式；
- (3) 比较不同变换域信息隐藏算法在嵌入前后的 DCT 系数直方图特征。

采用 JPEG 格式灰度图像作为载体图像，用 MATLAB 实现 JPEG 图片变换域信息隐藏的嵌入与提取算法。自选 JPEG 载体图像，嵌入内容自选。

1、要求实现 JSTEG 信息嵌入与提取算法。并比较嵌入前后的视觉效果与 DCT 系数直方图。

2、要求实现 F3 信息嵌入与提取算法。并比较嵌入前后的视觉效果与 DCT 系数直方图。

3、要求实现 F4 信息嵌入与提取算法。并比较嵌入前后的视觉效果与 DCT 系数直方图。

4、要求实现 F5 信息嵌入与提取算法。并比较嵌入前后的视觉效果与 DCT 系数直方图。

2.2 系统设计

考虑到 JSTEG 系列算法差别较小, 因此使用通用框架如下图：

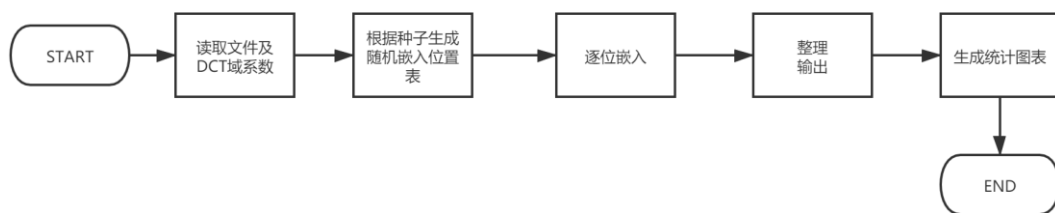


图 2-1 通用框架

其中读取文件 DCT 系数由工具执行. 下面主要描述嵌入部分的流程.

2.3 系统实现

2.3.1 行列解算

从随机数表中读取 index, 并根据图片尺寸解算为行向量和列向量.

```

1. location=randtable(RandTableIndex);
2. %Try to Resolve
3. row=ceil(location/400);
4. column=mod(location,400)+1;
    
```

2.3.2 Jsteg 算法

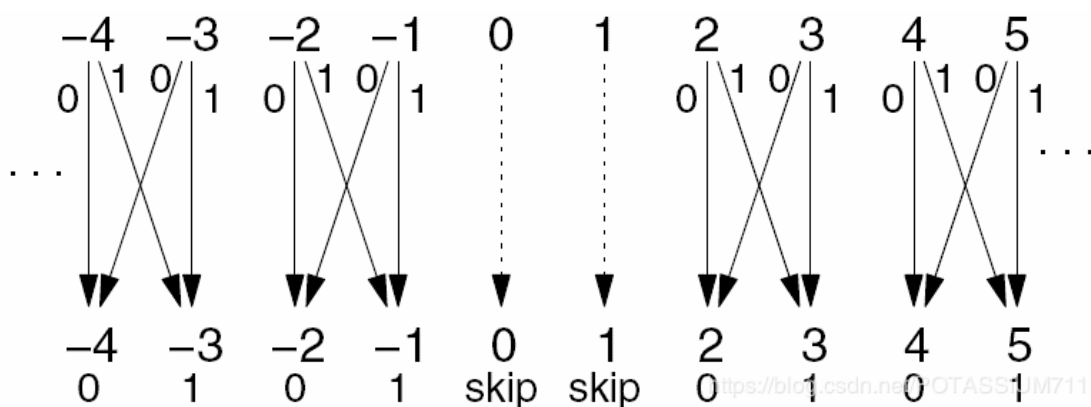


图 2-2 Jsteg 算法原理

读取系数和待嵌入比特流:

嵌入:

1. 0/1: 跳过
2. 通过保持或 $x \pm 1x \setminus pmlx \pm 1$ 进行调整以适合最低位数 (0-偶数, 1-奇数)
3. 2-1 不能达到 1, 因此必须更改为 3。

提取:

1. 0/1: 跳过
2. 只需读取最低位数 (0 偶, 1 奇)

2.3.3 F3 算法

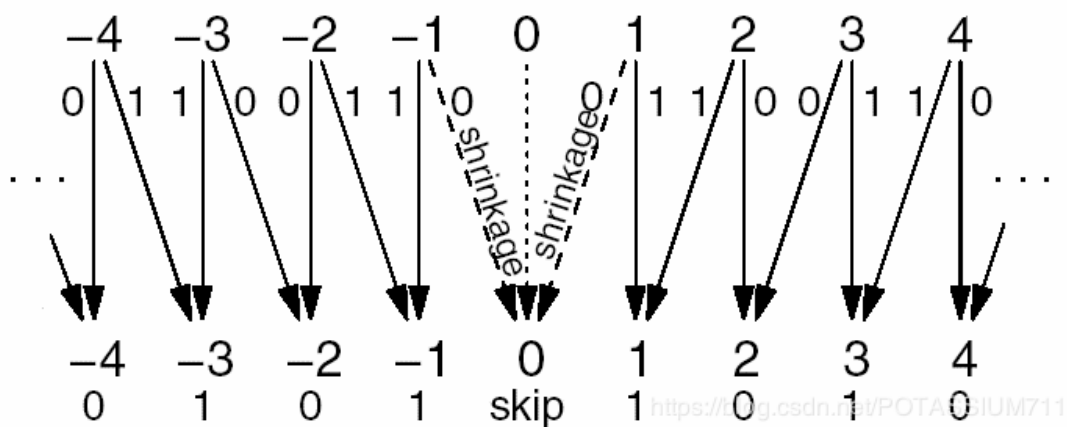


图 2-3 F3 算法原理

嵌入:

1. 0: 跳过
2. 正负部分是对称的。
3. 1 / -1 和 0: shrinkage
4. 通过保持或 $|x| - 1 = 1 \mid x \mid - 1 \mid x \mid - 1$ 进行调整以适合最低位数 (0-偶数, 1-奇数)

提取:

1. 0: 跳过
2. 只需读取最低位数 (0 偶, 1 奇)

2.3.4 F4 算法

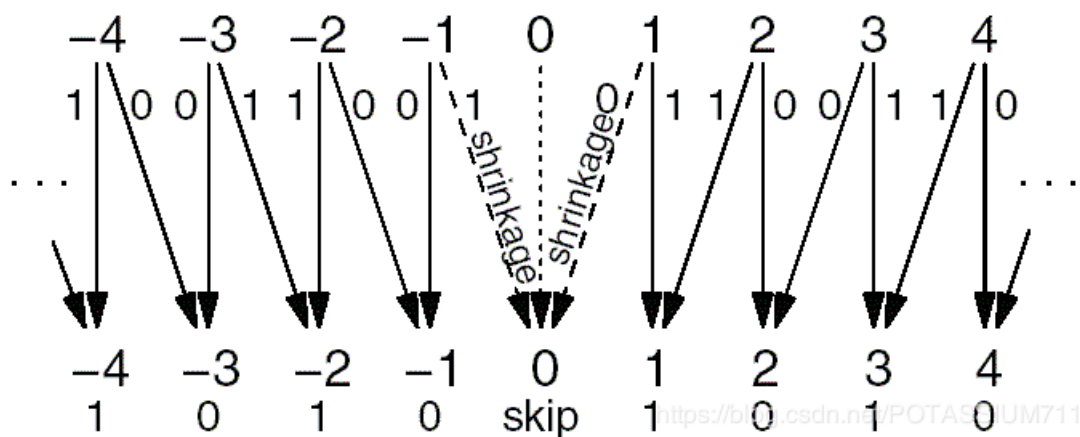


图 2-4 F4 算法

嵌入:

1. 0: 跳过
2. 正负部分是对应的。
3. 1-0 和 -1-1: shrinkage
4. 对于正数: 奇数 1, 偶数 0
5. 对于负数: 奇数 0, 偶数 1

提取:

1. 0: 跳过
2. 对于正数: 读取最低的数字
3. 对于负数: 读取最低的数字并反转

2.3.5 F5 算法

矩阵编码:

1. $\text{bool } p = (\text{if } x_1 == \text{lowbit}(a_1 \oplus a_3))$
2. $\text{bool } q = (\text{if } x_2 == \text{lowbit}(a_2 \oplus a_3))$
3. $p \ \&\& \ q ? \text{keep} : \text{pass};$
4. $!p \ \&\& \ q ? \text{Change } a_1 : \text{pass};$
5. $p \ \&\& \ !q ? \text{Change } a_2 : \text{pass};$
6. $!p \ \&\& \ !q ? \text{Change } a_3 : \text{pass};$

图 2-5 矩阵编码的逻辑描述

选择非 0 DCT 系数并洗牌

对于每 3 个 DCT 系数，尝试使用矩阵编码嵌入

[注意]此过程中的 shrinkage: 编码后，如果任何 DCT 系数变为 0，则取消此次嵌入并加载 3 个 DCT 系数，然后重新开始。

2.3.6 运行结果

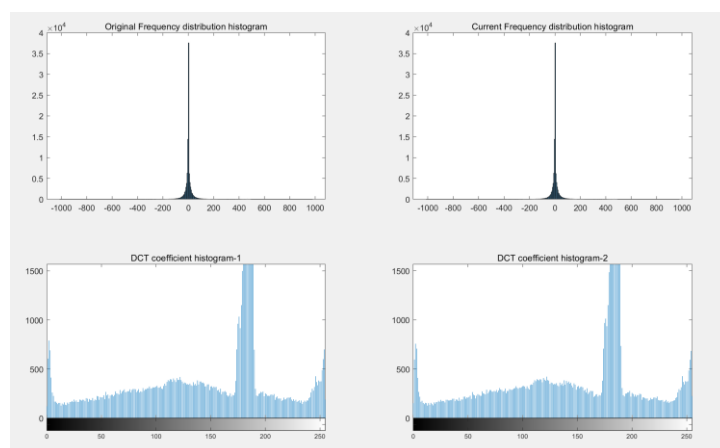


图 2-6 Jsteg 嵌入结果

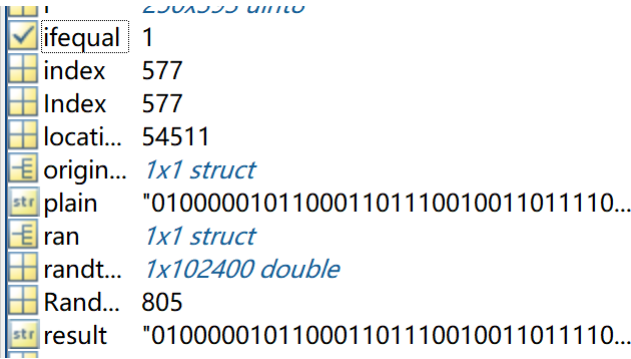


图 2-7 Jsteg 提取结果

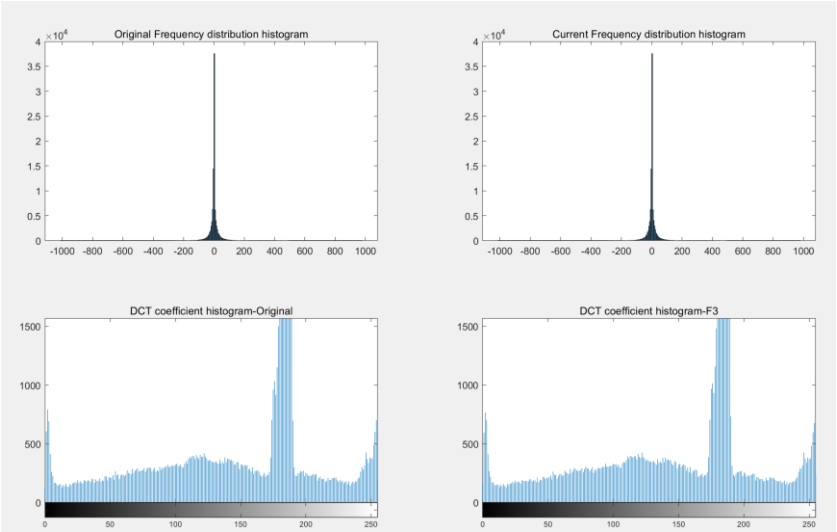


图 2-8 F3 嵌入

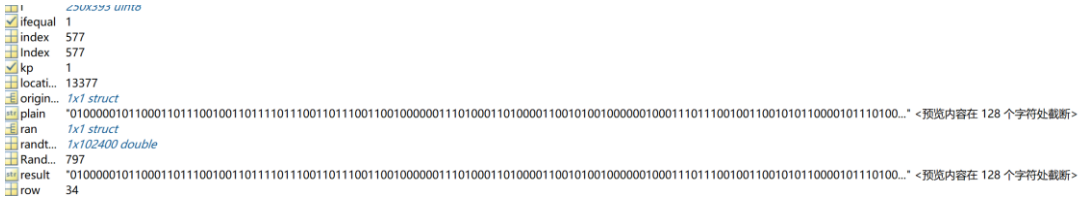


图 2-9 F3 提取

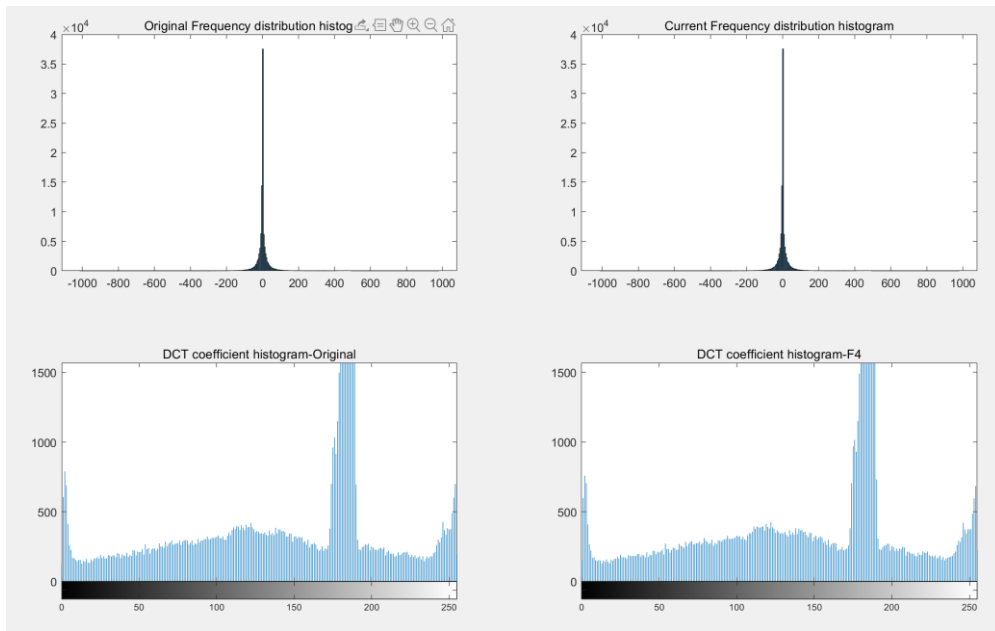


图 2-10 F4 嵌入

ifequal	1
ifequal	1
index	577
index	577
kp	1
locati...	51418
origin...	1x1 struct
plain	*0100000101100011011100100110111100110111001100100000011101000110100001100101001000000100011101110010011001010110000101110100...
ran	1x1 struct
randt...	1x102400 double
Rand...	799
result	*01000001011000110111001001101111001101110011001100100000011101000110100001100101001000000100011101110010011001010110000101110100...
row	129

图 2-11 F4 提取

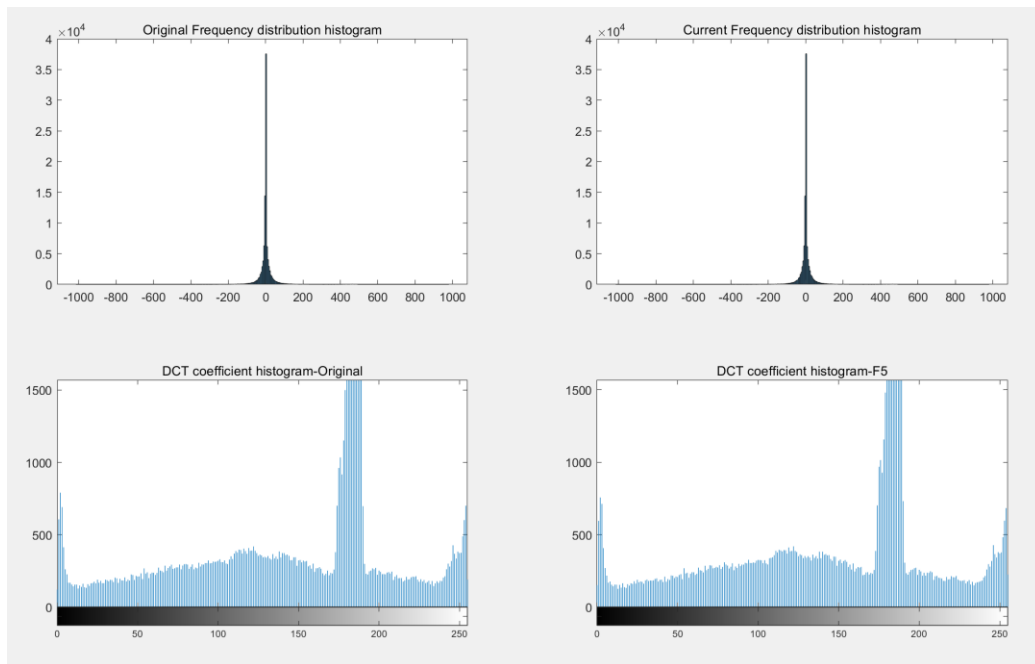


图 2-12 F5 嵌入

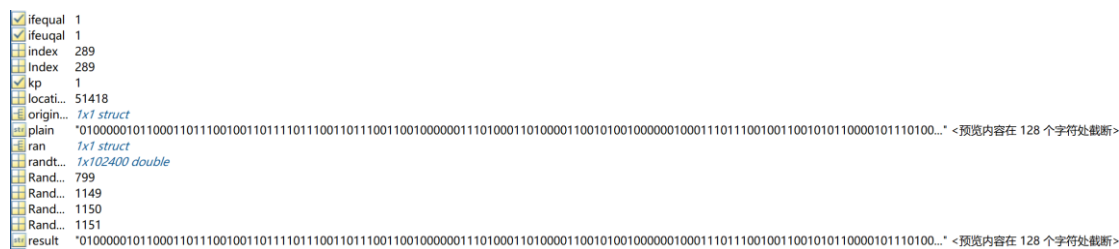


图 2-13 F5 提取

2.4 实验小结

本次实验主要是为了理解图片隐藏的两种算法, 算法比较简单, 但由于是第一次接触 matlab, 也花了不少时间摸索. 通过这次实验, 深刻理解了两种算法的过程并且逐步熟悉了 matlab 的使用.

个人原理总结已发在个人博客中:

<https://blog.csdn.net/POTASSIUM711/article/details/105052334>

代码已在 Github 上开源:

https://github.com/IRIDIUM-SUB/Steg_Experiment_HUST/tree/master/Jsteg

3 指导教师评定意见

一、对实验报告的评语

二、对实验报告评分

评分项目 (分值)	程序内容 (36.8 分)	程序规范 (9.2 分)	报告内容 (36.8 分)	报告规范 (9.2 分)	考勤 (8 分)	逾期扣分	合 计 (100 分)
得分							

4 附录 A 基于 LSB 的空域信息隐藏实现的源程序

4.1 嵌入

```
1. Picture=imread('kinkakuji01.bmp');
2. Double_Picture=Picture;
3. Double_Picture=double(Double_Picture);
4. %wen.txt_id=fopen('wen.txt','r');
5. %[msg,len]=fread(wen.txt_id,'ubit1');
6. msg='01000001011000110111001001101111011100110111001100100000011101000110100
0011001010010000001000111011100100110010101100001011101000010000001010111011
0000101101100011011000010000001010111011001010010000001100011011000010110111
0001000000101001001100101011000010110001101101000001000000100010101110110011
0010101110010011110010010000001000011011011110111001001101110011001010111001
0001000000110111101100110001000000111010001101000011001010010000001010111011
0111101110010011011000110010000100000010101010011001000110000001100010011011
10011000100110100001110000011100000110110';%Across the Great Wall We can Rea
ch Every Corner of the World U201714886
7. len=strlength(string(msg));
8. [m,n]=size(Double_Picture);
9. p=1;
10. for f2=1:n
11.
12.     for f1=1:m
13.         fprintf("[LOG] Embedding %d %d\t%d\n",f1,f2,Double_Picture(f1,f2));
14.         Double_Picture(f1,f2)=Double_Picture(f1,f2)-
            mod(Double_Picture(f1,f2),2)+msg(p);
15.         if p==len
16.             break;
17.         end
18.         p=p+1;
19.     end
```



```

20.     if p==len
21.         break;
22.     end
23. end
24. Double_Picture=uint8(Double_Picture);
25. imwrite(Double_Picture,'kinkakuji01-steg.bmp');
26. subplot(121);imshow(Picture);title('Original Pic');
    27. subplot(122);imshow(Double_Picture);title('Embedded Pic');

```

4.2 提取

```

    1. Picture=imread('kinkakuji01-steg.bmp');
2. Picture=double(Picture);
3. [m,n]=size(Picture);
4. result="";
5. len=568;
6. p=1;
7. for f2=1:n
8.     for f1=1:m
9.         if bitand(Picture(f1,f2),1)==1
10.
11.             result=result+"1";
12.         else
13.
14.             result=result+"0";
15.         end
16.         if p==len
17.             break;
18.         end
19.         if p<len
20.             p=p+1;
21.         end
22.     end
23.     if p==len
24.         break;
25.     end
26. end
    27. fprintf("[SUCCESS] Msg=%s\n",result);

```

5 附录 B JPEG 图像变换域信息隐藏实现的源程序

5.1 Jsteg 嵌入

```

1.  A=imread('kinkakuji01.jpg');
2.  I=rgb2gray(A); %RGBÍ¼x³»`Î³»ÒŒËÍ¼
3.  imwrite(I,'kinkakuji01g.jpg','quality',100);
4.  originobj= jpeg_read('kinkakuji01g.jpg');
5.  activedobj=originobj;
6.  subplot(2,2,1)
7.  histogram(originobj.coef_arrays{1,1});%display Frequency distribution histogram
8.  title("Original Frequency distribution histogram");
9.  plain='010000010110001101110010011011110111001101110011001000000111010001101
0000110010100100000010001110111001001100101011000010111010000100000010101110
1100001011011000110110000100000010101110110010100100000011000110110000101101
1100010000001010010011001010110000101100011011010000010000001000101011101100
1100101011100100111100100100000010000110110111101110010011011100110010101110
0100010000001101111011001100010000001110100011010000110010100100000010101110
1101111011100100110110001100100001000000101010100110010001100000011000100110
1110011000100110100001110000011100000110110';%Across the Great Wall We can R
each Every Corner of the World U201714886
10. ran=rng(7742085);%Seed No repeat in this seed
11.
12. %DCT Table's Size=256*400
13. randtable=randperm(256*400);
14. FrequencyStatic=tabulate(randtable(:));
15. RandTableIndex=1;
16. index=1;
17. while index<=length(plain)
18.     %Get Location
19.     location=randtable(RandTableIndex);
20.     %Try to Resolve
21.     row=ceil(location/400);
22.     column=mod(location,400)+1;
23.     %Try to Replace

```

```

24.     if activedobj.coef_arrays{1,1}(row,column)==0||activedobj.coef_arrays{1,
    1}(row,column)==1
25.         %Fail to Fetch, Throw
26.         RandTableIndex=RandTableIndex+1;%Fetch the Next One
27.         continue
28.     end
29.
30.     if plain(index)=='1'
31.
32.         if mod(activedobj.coef_arrays{1,1}(row,column),2)==0%Even
33.
34.             activedobj.coef_arrays{1,1}(row,column)=activedobj.coef_arrays{1,
    1}(row,column)+1;%eg. -4->-3,2->3
35.
36.             %fprintf("1,+1");
37.             end
38.             %Odd Number Won't Be Changed
39.             index=index+1;
40.             %Debug
41.             fprintf("1-
    RandTableIndex%d\tLocation%d\n",RandTableIndex,randtable(RandTableIndex));
42.             RandTableIndex=RandTableIndex+1;
43.             continue%Warning: Unknown Bug, Patched temporarily
44.         end
45.
46.         if plain(index)=='0'
47.             location=randtable(RandTableIndex);%Unknown bug occurred. Patch it t
    emporarily
48.             if mod(activedobj.coef_arrays{1,1}(row,column),2)==1%Odd
49.                 activedobj.coef_arrays{1,1}(row,column)=activedobj.coef_arrays{1,
    1}(row,column)-1;%eg.-3->-4,3->2
50.                 %fprintf("0,-1");
51.                 end
52.                 %Even Number Won't Be Changed
53.                 index=index+1;
54.                 %Debug
55.                 fprintf("0-
    RandTableIndex%d\tLocation%d\n",RandTableIndex,randtable(RandTableIndex));
56.                 RandTableIndex=RandTableIndex+1;
57.             end
58.

```

```

59. end
60. subplot(2,2,2)
61. histogram(activatedobj.coef_arrays{1,1});%display Frequency distribution histogram
62. title("Current Frequency distribution histogram");
63.
64. result=isequal(originobj.coef_arrays{1,1},activatedobj.coef_arrays{1,1});
65. jpeg_write(activatedobj,'kinkakuji-Jsteg.jpg');
66. %Show DCT coefficient histogram
67. Aft=imread('kinkakuji-Jsteg.jpg');
68. subplot(2,2,3)
69. imhist(I);
70. title("DCT coefficient histogram-1");
71.
72. subplot(2,2,4)
73. imhist(Aft);
74. title("DCT coefficient histogram-2");

```

5.2 Jsteg 提取

```

1. %Read image
2. originobj= jpeg_read('kinkakuji-Jsteg.jpg');
3. I=imread('kinkakuji-Jsteg.jpg');
4. activatedobj=originobj;
5.
6. ran=rng(7742085);%Seed No repeat in this seed
7. result="";%Store 0-1 string
8. randtable=randperm(256*400);%Shuffle
9. FrequencyStatic=tabulate(randtable(:));
10. RandTableIndex=1;
11. Index=1;
12.
13. %Debug
14.
15.
16. while Index<=576%Len of plain
17.     %Get Location
18.     location=randtable(RandTableIndex);
19.     %Try to Resolve
20.     row=ceil(location/400);

```

```

21.     column=mod(location,400)+1;
22.     %Try
23.     if activedobj.coef_arrays{1,1}(row,column)==0||activedobj.coef_arrays{1,
        1}(row,column)==1
24.         %Fail to Fetch, Throw
25.         RandTableIndex=RandTableIndex+1;%Fetch the Next One
26.         continue
27.     end
28.     %For any vavild cell
29.     steg=activedobj.coef_arrays{1,1}(row,column);
30.     %Debug
31.
32.     fprintf("%d-%d-
        RandTableIndex%d\tLocation%d\n",steg,mod(steg,2),RandTableIndex,location);
33.     %Recover
34.     %steg=odd->1
35.     %steg=even->0
36.     result=result+int2str(mod(steg,2));
37.
38.     RandTableIndex=RandTableIndex+1;
39.     Index=Index+1;
40. end
41. 9. plain='01000001011000110111001001101111011100110111001100100000011101000
        1101000011001010010000001000111011100100110010101100001011101000010000001010
        1110110000101101100011011000010000001010111011001010010000001100011011000010
        1101110001000000101001001100101011000010110001101101000001000000100010101110
        1100110010101110010011110010010000001000011011011110111001001101110011001010
        1110010001000000110111101100110001000000111010001101000011001010010000001010
        1110110111101110010011011000110010000100000010101010011001000110000001100010
        01101110011000100110100001110000011100000110110';%Across the Great Wall We c
        an Reach Every Corner of the World U201714886
42. ifequal=isequal(plain,result);%Final Comparison
    
```

5.3 F3 嵌入

```

1. A=imread('kinkakuji01.jpg');
2. I=rgb2gray(A); %RGBÍ¼«»`Î»»ÒŒÉÍ¼
3. imwrite(I,'kinkakuji01g.jpg','quality',100);
4. originobj= jpeg_read('kinkakuji01g.jpg');
5. activedobj=originobj;
    
```

```

6. subplot(2,2,1)
7.
8.
9. histogram(originobj.coef_arrays{1,1});%display Frequency distribution histogram
10. title("Original Frequency distribution histogram");
11. plain='01000001011000110111001001101111011100110111001100100000011101000110
1000011001010010000001000111011100100110010101100001011101000010000001010111
0110000101101100011011000010000001010111011001010010000001100011011000010110
11100010000001010010011001011100001011000110110100001000000100010101110110
0110010101110010011110010010000001000011011011110111001001101110011001010111
0010001000000110111101100110001000000111010001101000011001010010000001010111
0110111101110010011011000110010000100000010101010011001000110000001100010011
01110011000100110100001110000011100000110110';%Across the Great Wall We can
Reach Every Corner of the World U201714886
12. ran=rng(7742085);%Seed No repeat in this seed
13.
14. %DCT Table's Size=256*400
15. randtable=randperm(256*400);
16. FrequencyStatic=tabulate(randtable(:));
17. RandTableIndex=1;
18. index=1;
19. while index<=length(plain)
20.     %Get Location
21.     location=randtable(RandTableIndex);
22.     %Try to Resolve
23.     row=ceil(location/400);
24.     column=mod(location,400)+1;
25.     %Try to Replace
26.     if activedobj.coef_arrays{1,1}(row,column)==0
27.         %Fail to Fetch, Throw
28.         fprintf("Fail to fetch,%d\t%d\n",index,RandTableIndex);
29.         RandTableIndex=RandTableIndex+1;%Fetch the Next One
30.
31.         continue
32.     end
33.     % if plain=1 and even->abs-1
34.     % if plain=0 and odd->abs-1
35.     % if plain=1 and odd->Keep
36.     % if plain=0 and even->Keep
37.     % if plain=0 and 1/-1->Pass

```

```

38.     if plain(index)=='1'
39.         if mod(activatedobj.coef_arrays{1,1}(row,column),2)==0
40.             if activatedobj.coef_arrays{1,1}(row,column)>0
41.                 activatedobj.coef_arrays{1,1}(row,column)=activatedobj.coef_arrays{1,1}(row,column)-1;
42.             else
43.                 activatedobj.coef_arrays{1,1}(row,column)=activatedobj.coef_arrays{1,1}(row,column)+1;
44.             end
45.             fprintf("1-
RandTableIndex%d\tLocation%d\n",RandTableIndex,randtable(RandTableIndex));
46.         end
47.         index=index+1;
48.         RandTableIndex=RandTableIndex+1;
49.     else
50.         if activatedobj.coef_arrays{1,1}(row,column)==1||activatedobj.coef_arrays{1,1}(row,column)==-1
51.             %Pass
52.             %Shrink to 0
53.             activatedobj.coef_arrays{1,1}(row,column)=0;
54.             RandTableIndex=RandTableIndex+1;
55.             fprintf("(1/-
1,0),Fail to fetch,%d\t%d\n",RandTableIndex,randtable(RandTableIndex));
56.             continue;
57.         end
58.         if mod(activatedobj.coef_arrays{1,1}(row,column),2)==1
59.             if activatedobj.coef_arrays{1,1}(row,column)>0
60.                 activatedobj.coef_arrays{1,1}(row,column)=activatedobj.coef_arrays{1,1}(row,column)-1;
61.             else
62.                 activatedobj.coef_arrays{1,1}(row,column)=activatedobj.coef_arrays{1,1}(row,column)+1;
63.             end
64.
65.         end
66.         fprintf("0-
RandTableIndex%d\tLocation%d\n",RandTableIndex,randtable(RandTableIndex));
67.         index=index+1;
68.         RandTableIndex=RandTableIndex+1;
69.
70.     end

```

```

71. end
72. subplot(2,2,2)
73. histogram(activatedobj.coef_arrays{1,1});%display Frequency distribution histogram
74. title("Current Frequency distribution histogram");
75.
76. result=isequal(originobj.coef_arrays{1,1},activatedobj.coef_arrays{1,1});
77. jpeg_write(activatedobj,'kinkakuji-F3.jpg');
78. %Show DCT coefficient histogram
79. Aft=imread('kinkakuji-F3.jpg');
80. subplot(2,2,3)
81. imhist(I);
82. title("DCT coefficient histogram-Original");
83.
84. subplot(2,2,4)
85. imhist(Aft);
86. title("DCT coefficient histogram-F3");

```

5.4 F3 提取

```

1. %Read image
2. originobj= jpeg_read('kinkakuji-F3.jpg');
3. I=imread('kinkakuji-Jsteg.jpg');
4. activatedobj=originobj;
5. ran=rng(7742085);%Seed No repeat in this seed
6. result="";%Store 0-1 string
7. randtable=randperm(256*400);%Shuffle
8. FrequencyStatic=tabulate(randtable(:));
9. RandTableIndex=1;
10. Index=1;
11.
12. %Debug
13.
14.
15. while Index<=576%Len of plain
16.     %Get Location
17.     location=randtable(RandTableIndex);
18.     %Try to Resolve
19.     row=ceil(location/400);
20.     column=mod(location,400)+1;

```



```

21.    %Try
22.    if activedobj.coef_arrays{1,1}(row,column)==0
23.        %Fail to Fetch, Throw
24.        RandTableIndex=RandTableIndex+1;%Fetch the Next One
25.        continue
26.    end
27.    %For any vavild cell
28.    steg=activedobj.coef_arrays{1,1}(row,column);
29.    %Debug
30.
31.    fprintf("%d-%d-
    RandTableIndex%d\tLocation%d\n",steg,mod(steg,2),RandTableIndex,location);
32.    %Recover
33.    %steg=odd->1
34.    %steg=even->0
35.    result=result+int2str(mod(steg,2));
36.
37.    RandTableIndex=RandTableIndex+1;
38.    Index=Index+1;
39. end
40. plain='01000001011000110111001001101111011100110111001100100000011101000110
    1000011001010010000001000111011100100110010101100001011101000010000001010111
    0110000101101100011011000010000001010111011001010010000001100011011000010110
    1110001000000101001001100101011000010110001101101000001000000100010101110110
    0110010101110010011110010010000001000011011011110111001001101110011001010111
    0010001000000110111101100110001000000111010001101000011001010010000001010111
    0110111101110010011011000110010000100000010101010011001000110000001100010011
    01110011000100110100001110000011100000110110';%Across the Great Wall We can
    Reach Every Corner of the World U201714886
41.    kp=isequal(plain,result);%Final Comparison

```

5.5 F4 嵌入

```

1.    A=imread('kinkakuji01.jpg');
2.    I=rgb2gray(A); %RGBÍ¼x³»~Î³»ÒŒÈÍ¼
3.    imwrite(I,'kinkakuji01g.jpg','quality',100);
4.    originobj= jpeg_read('kinkakuji01g.jpg');
5.    activedobj=originobj;
6.    subplot(2,2,1)
7.

```

```

8.
9. histogram(originobj.coef_arrays{1,1});%display Frequency distribution histogram
10. title("Original Frequency distribution histogram");
11. plain='010000010110001101110010011011110111001101110011001000000111010001101
0000110010100100000010001110111001001100101011000010111010000100000010101110
1100001011011000110110000100000010101110110010100100000011000110110000101101
110001000000101001001100101011000010110001101101000010000001000101011101100
1100101011100100111100100100000010000110110111101110010011011100110010101110
0100010000001101111011001100010000001110100011010000110010100100000010101110
1101111011100100110110001100100001000000101010100110010001100000011000100110
1110011000100110100001110000011100000110110';%Across the Great Wall We can R
each Every Corner of the World U201714886
12. ran=rng(7742085);%Seed No repeat in this seed
13. uosedindex=[];
14. throwedindex=[];
15. %DCT Table's Size=256*400
16. randtable=randperm(256*400);
17. FrequencyStatic=tabulate(randtable(:));
18. RandTableIndex=1;
19. index=1;
20. while index<=length(plain)
21.     %Get Location
22.     location=randtable(RandTableIndex);
23.     %Try to Resolve
24.     row=ceil(location/400);
25.     column=mod(location,400)+1;
26.     %Try to Replace
27.     if activedobj.coef_arrays{1,1}(row,column)==0
28.         %Fail to Fetch, Throw
29.         fprintf("Fail to fetch,%d\t%d\n",index,RandTableIndex);
30.         RandTableIndex=RandTableIndex+1;%Fetch the Next One
31.
32.         continue
33.     end
34.     % if plain=1 and positive odd->keep
35.     % if plain=1 and positive even->abs-1
36.     % if plain=1 and negative odd->abs-1
37.     % if plain=1 and negative even->keep
38.     %
39.     % if plain=0 and positive odd->abs-1

```

```

40.    % if plain=0 and positive even->keep
41.    % if plain=0 and negative odd->keep
42.    % if plain=0 and negative even->abs-1
43.    %
44.    % if plain=0 and 1->Shrinkage
45.    % if plain=1 and -1->Shrinkage
46.    if plain(index)=='1'
47.        if activedobj.coef_arrays{1,1}(row,column)==-1
48.            %Pass
49.            %Shrink to 0
50.            activedobj.coef_arrays{1,1}(row,column)=0;
51.            RandTableIndex=RandTableIndex+1;
52.            fprintf("(-
1,1),Fail to fetch,%d\t%d\n",RandTableIndex,randtable(RandTableIndex));
53.            continue;
54.        end
55.
56.        if (mod(activedobj.coef_arrays{1,1}(row,column),2)==0&&activedobj.co
ef_arrays{1,1}(row,column)>0) || (mod(activedobj.coef_arrays{1,1}(row,column),
2)~=0&&activedobj.coef_arrays{1,1}(row,column)<0)
57.            if activedobj.coef_arrays{1,1}(row,column)>0
58.                activedobj.coef_arrays{1,1}(row,column)=activedobj.coef_arra
ys{1,1}(row,column)-1;
59.            else
60.                activedobj.coef_arrays{1,1}(row,column)=activedobj.coef_arra
ys{1,1}(row,column)+1;
61.            end
62.        end
63.
64.        fprintf("1-
RandTableIndex%d\tLocation%d\n",RandTableIndex,randtable(RandTableIndex));
65.        index=index+1;
66.        RandTableIndex=RandTableIndex+1;
67.    else
68.        if activedobj.coef_arrays{1,1}(row,column)==1
69.            %Pass
70.            %Shrink to 0
71.            activedobj.coef_arrays{1,1}(row,column)=0;
72.            RandTableIndex=RandTableIndex+1;
73.            fprintf("(1,0),Fail to fetch,%d\t%d\n",RandTableIndex,randtable
(RandTableIndex));

```

```

74.         continue;
75.     end
76.     if (mod(activatedobj.coef_arrays{1,1}(row,column),2)~=0&&activatedobj.c
        oef_arrays{1,1}(row,column)>0) || (mod(activatedobj.coef_arrays{1,1}(row,column)
        ,2)==0&&activatedobj.coef_arrays{1,1}(row,column)<0)
77.         if activatedobj.coef_arrays{1,1}(row,column)>0
78.             activatedobj.coef_arrays{1,1}(row,column)=activatedobj.coef_arra
            ys{1,1}(row,column)-1;
79.         else
80.             activatedobj.coef_arrays{1,1}(row,column)=activatedobj.coef_arra
            ys{1,1}(row,column)+1;
81.         end
82.
83.     end
84.     fprintf("0-
        RandTableIndex%d\tLocation%d\n",RandTableIndex,randtable(RandTableIndex));
85.     index=index+1;
86.     RandTableIndex=RandTableIndex+1;
87.
88. end
89. end
90. subplot(2,2,2)
91. histogram(activatedobj.coef_arrays{1,1});%display Frequency distribution histo
    gram
92. title("Current Frequency distribution histogram");
93.
94. result=isequal(originobj.coef_arrays{1,1},activatedobj.coef_arrays{1,1});
95. jpeg_write(activatedobj,'kinkakuji-F4.jpg');
96. %Show DCT coefficient histogram
97. Aft=imread('kinkakuji-F4.jpg');
98. subplot(2,2,3)
99. imhist(I);
100. title("DCT coefficient histogram-Original");
101.
102. subplot(2,2,4)
103. imhist(Aft);
    104. title("DCT coefficient histogram-F4");

```

5.6 F4 提取

```

1. %Read image
2. originobj= jpeg_read('kinkakuji-F4.jpg');
3. I=imread('kinkakuji-Jsteg.jpg');
4. activedobj=originobj;
5. ran=rng(7742085);%Seed No repeat in this seed
6. result="";%Store 0-1 string
7. randtable=randperm(256*400);%Shuffle
8. FrequencyStatic=tabulate(randtable(:));
9. RandTableIndex=1;
10. Index=1;
11.
12. %Debug
13.
14.
15.
16.
17. while Index<=576%Len of plain
18.     %Get Location
19.     location=randtable(RandTableIndex);
20.     %Try to Resolve
21.     row=ceil(location/400);
22.     column=mod(location,400)+1;
23.     %Try
24.     if activedobj.coef_arrays{1,1}(row,column)==0
25.         %Fail to Fetch, Throw
26.         RandTableIndex=RandTableIndex+1;%Fetch the Next One
27.         continue
28.     end
29.     %For any vavild cell
30.     steg=activedobj.coef_arrays{1,1}(row,column);
31.     %Debug
32.     %usedindex(end+1)=location;
33.
34.     %Recover
35.     %positive odd and negative even->1
36.     %positive even and negative odd->0
37.     if (mod(steg,2)&&steg>0)|| (mod(steg,2)==0&&steg<0)
38.         result=result+"1";

```

```

39.         fprintf("%d-1-
RandTableIndex%d\tLocation%d\n",steg,RandTableIndex,location);
40.     else
41.         result=result+"0";
42.         fprintf("%d-0-
RandTableIndex%d\tLocation%d\n",steg,RandTableIndex,location);
43.     end
44.
45.
46.     RandTableIndex=RandTableIndex+1;
47.     Index=Index+1;
48. end
49. plain='010000010110001101110010011011110111001101110011001000000111010001101
0000110010100100000010001110111001001100101011000010111010000100000010101110
1100001011011000110110000100000010101110110010100100000011000110110000101101
110001000000101001001100101011000010110001101101000010000001000101011101100
1100101011100100111100100100000010000110110111101110010011011100110010101110
0100010000001101111011001100010000001110100011010000110010100100000010101110
1101111011100100110110001100100001000000101010100110010001100000011000100110
1110011000100110100001110000011100000110110';%Across the Great Wall We can R
each Every Corner of the World U201714886
50.
51. ifequal=isequal(plain,result);%Final Comparison

```

5.7 F5 嵌入

```

1. A=imread('kinkakuji01.jpg');
2. I=rgb2gray(A); %RGBÍ%×»~Î»»ÒŒÍ%
3. imwrite(I,'kinkakuji01g.jpg','quality',100);
4. originobj= jpeg_read('kinkakuji01g.jpg');
5. activedobj=originobj;
6. subplot(2,2,1);
7.
8.
9. histogram(originobj.coef_arrays{1,1});%display Frequency distribution histog
ram
10. title("Original Frequency distribution histogram");
11. plain='010000010110001101110010011011110111001101110011001000000111010001101
0000110010100100000010001110111001001100101011000010111010000100000010101110
1100001011011000110110000100000010101110110010100100000011000110110000101101

```

```

1100010000001010010011001010110000101100011011010000010000001000101011101100
1100101011100100111100100100000010000110110111101110010011011100110010101110
0100010000001101111011001100010000001110100011010000110010100100000010101110
1101111011100100110110001100100001000000101010100110010001100000011000100110
1110011000100110100001110000011100000110110';%Across the Great Wall We can R
each Every Corner of the World U201714886
12.
13. ran=rng(7742085);%Seed No repeat in this seed
14. uosedindex=[];
15. throwedindex=[];
16. %DCT Table's Size=256*400
17. randtable=randperm(256*400);
18. FrequencyStatic=tabulate(randtable(:));
19. RandTableIndex1=1;
20. RandTableIndex2=2;
21. RandTableIndex3=3;
22. index=1;
23.
24. % For Each 3 DCT coef and 2 plain
25. while index<=length(plain)/2
26.
27.     a1=randtable(RandTableIndex1);
28.     row1=ceil(a1/400);
29.     column1=mod(a1,400)+1;
30.
31.     a2=randtable(RandTableIndex2);
32.     row2=ceil(a2/400);
33.     column2=mod(a2,400)+1;
34.
35.     a3=randtable(RandTableIndex3);
36.     row3=ceil(a3/400);
37.     column3=mod(a3,400)+1;
38.
39.
40.     x1=str2num(plain((index-1)*2+1));
41.     x2=str2num(plain((index-1)*2+2));
42.
43.     %Check and Drop
44.     %Pretty Slow But Readable
45.     if activedobj.coef_arrays{1,1}(row1,column1)==0
46.         RandTableIndex1=RandTableIndex1+1;

```

```

47.         RandTableIndex2=RandTableIndex2+1;
48.         RandTableIndex3=RandTableIndex3+1;
49.         fprintf("[LOG] a1=0, move,%d\t%d\t%d\n",RandTableIndex1,RandTableIndex2,RandTableIndex3);
50.         continue;%Restart
51.     else
52.         if activedobj.coef_arrays{1,1}(row2,column2)==0
53.             RandTableIndex2=RandTableIndex2+1;
54.             RandTableIndex3=RandTableIndex3+1;
55.             fprintf("[LOG] a2=0, move,%d\t%d\t%d\n",RandTableIndex1,RandTableIndex2,RandTableIndex3);
56.             continue;%Restart
57.         else
58.             if activedobj.coef_arrays{1,1}(row3,column3)==0
59.                 RandTableIndex3=RandTableIndex3+1;
60.                 fprintf("[LOG] a3=0, move,%d\t%d\t%d\n",RandTableIndex1,RandTableIndex2,RandTableIndex3);
61.                 continue;%Restart
62.             end
63.         end
64.     end
65.
66.     %Now it should be 3 non-0 DCT coef
67.     fprintf("[READY] Current DCT Coef Group:%d\t%d\t%d\n",activedobj.coef_arrays{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,column3));
68.     if mod(bitxor(activedobj.coef_arrays{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row3,column3),'int16'),2)==x1&&mod(bitxor(activedobj.coef_arrays{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,column3),'int16'),2)==x2%p && q?keep:pass;
69.         fprintf("[LOG] No Change,%d\t%d\t%d \t %d\t%d\n",activedobj.coef_arrays{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,column3),x1,x2);
70.     else
71.         if mod(bitxor(activedobj.coef_arrays{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row3,column3),'int16'),2)~=x1&&mod(bitxor(activedobj.coef_arrays{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,column3),'int16'),2)~=x2%!p&&q?Change a1:pass;
72.             fprintf("[LOG] Change a1,%d\t%d\t%d \t %d\t%d\n",activedobj.coef_arrays{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,column3),x1,x2);

```



```

73.         if activedobj.coef_arrays{1,1}(row1,column1)>0
74.             activedobj.coef_arrays{1,1}(row1,column1)=activedobj.coef_arrays{1,1}(row1,column1)-1;
75.         else
76.             activedobj.coef_arrays{1,1}(row1,column1)=activedobj.coef_arrays{1,1}(row1,column1)+1;
77.         end
78.     else
79.         if mod(bitxor(activedobj.coef_arrays{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row3,column3),'int16'),2)==x1&&mod(bitxor(activedobj.coef_arrays{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,column3),'int16'),2)~=x2%p&&!q?Change a2:pass;
80.             fprintf("[LOG] Change a2,%d\t%d\t%d \t %d\t%d\n",activedobj.coef_arrays{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,column3),x1,x2);
81.             if activedobj.coef_arrays{1,1}(row2,column2)>0
82.                 activedobj.coef_arrays{1,1}(row2,column2)=activedobj.coef_arrays{1,1}(row2,column2)-1;
83.             else
84.                 activedobj.coef_arrays{1,1}(row2,column2)=activedobj.coef_arrays{1,1}(row2,column2)+1;
85.             end
86.         else
87.             if mod(bitxor(activedobj.coef_arrays{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row3,column3),'int16'),2)~=x1&&mod(bitxor(activedobj.coef_arrays{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,column3),'int16'),2)~=x2!p&&!q?Change a3:pass;
88.                 fprintf("[LOG] Change a3,%d\t%d\t%d \t %d\t%d\n",activedobj.coef_arrays{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,column3),x1,x2);
89.                 if activedobj.coef_arrays{1,1}(row3,column3)>0
90.                     activedobj.coef_arrays{1,1}(row3,column3)=activedobj.coef_arrays{1,1}(row3,column3)-1;
91.                 else
92.                     activedobj.coef_arrays{1,1}(row3,column3)=activedobj.coef_arrays{1,1}(row3,column3)+1;
93.                 end
94.             end
95.         end
96.     end
97. end

```

```

98.
99.     %Shrinkage
100.     if activedobj.coef_arrays{1,1}(row1,column1)==0
101.         fprintf("[LOG] %d\tShrinkage a1,%d\t%d\t%d\n",index,RandTableIndex1
            ,RandTableIndex2,RandTableIndex3);
102.         RandTableIndex1=RandTableIndex2;
103.         RandTableIndex2=RandTableIndex3;
104.         RandTableIndex3=RandTableIndex3+1;
105.         continue;
106.     else
107.         if activedobj.coef_arrays{1,1}(row2,column2)==0
108.             fprintf("[LOG] %d\tShrinkage a2,%d\t%d\t%d\n",index,RandTableIndex1,RandTableIndex2,RandTableIndex3);
109.             RandTableIndex2=RandTableIndex3;
110.             RandTableIndex3=RandTableIndex3+1;
111.             continue;
112.         else
113.             if activedobj.coef_arrays{1,1}(row3,column3)==0
114.                 fprintf("[LOG] %d\tShrinkage a3,%d\t%d\t%d\n",index,RandTableIndex1,RandTableIndex2,RandTableIndex3);
115.                 RandTableIndex3=RandTableIndex3+1;
116.                 continue;
117.             end
118.         end
119.     end
120.     %Adjust DCT Coef Index
121.     %Start Over
122.     RandTableIndex1=RandTableIndex3+1;
123.     RandTableIndex2=RandTableIndex1+1;
124.     RandTableIndex3=RandTableIndex2+1;
125.     index=index+1;
126.     fprintf("[SUCCESS] Embedding Success:Next Group:%d\t%d\t%d\t%d\n",RandTableIndex1,RandTableIndex2,RandTableIndex3,index);
127. end
128. subplot(2,2,2)
129. histogram(activedobj.coef_arrays{1,1});%display Frequency distribution histogram
130. title("Current Frequency distribution histogram");
131.
132. result=isequal(originobj.coef_arrays{1,1},activedobj.coef_arrays{1,1});
133. jpeg_write(activedobj,'kinkakuji-F5.jpg');

```

```

134. %Show DCT coefficient histogram
135. Aft=imread('kinkakuji-F5.jpg');
136. subplot(2,2,3)
137. imhist(I);
138. title("DCT coefficient histogram-Original");
139.
140. subplot(2,2,4)
141. imhist(Aft);
    142. title("DCT coefficient histogram-F5");

```

5.8 F5 提取

```

1. %Read image
2. originobj= jpeg_read('kinkakuji-F5.jpg');
3. I=imread('kinkakuji-Jsteg.jpg');
4. activedobj=originobj;
5. ran=rng(7742085);%Seed No repeat in this seed
6. result="";%Store 0-1 string
7. randtable=randperm(256*400);%Shuffle
8. FrequencyStatic=tabulate(randtable(:));
9. RandTableIndex1=1;
10. RandTableIndex2=2;
11. RandTableIndex3=3;
12. Index=1;
13.
14. %Debug
15.
16. while Index<=576/2%Len of plain
17.     a1=randtable(RandTableIndex1);
18.     row1=ceil(a1/400);
19.     column1=mod(a1,400)+1;
20.
21.     a2=randtable(RandTableIndex2);
22.     row2=ceil(a2/400);
23.     column2=mod(a2,400)+1;
24.
25.     a3=randtable(RandTableIndex3);
26.     row3=ceil(a3/400);
27.     column3=mod(a3,400)+1;
28.

```

```

29.     if activedobj.coef_arrays{1,1}(row1,column1)==0
30.         RandTableIndex1=RandTableIndex1+1;
31.         RandTableIndex2=RandTableIndex2+1;
32.         RandTableIndex3=RandTableIndex3+1;
33.         fprintf("[LOG] a1=0, move,%d-%d-%d\n",RandTableIndex1,RandTableIndex
2,RandTableIndex3);
34.         continue;%Restart
35.     else
36.         if activedobj.coef_arrays{1,1}(row2,column2)==0
37.             RandTableIndex2=RandTableIndex2+1;
38.             RandTableIndex3=RandTableIndex3+1;
39.             fprintf("[LOG] a2=0, move,%d-%d-%d\n",RandTableIndex1,RandTableIndex
2,RandTableIndex3);
40.             continue;%Restart
41.         else
42.             if activedobj.coef_arrays{1,1}(row3,column3)==0
43.                 RandTableIndex3=RandTableIndex3+1;
44.                 fprintf("[LOG] a3=0, move,%d-%d-%d\n",RandTableIndex1,RandTa
bleIndex2,RandTableIndex3);
45.                 continue;%Restart
46.             end
47.         end
48.     end
49.     %Now it should be 3 non-0 DCT coef
50.     fprintf("[READY] Current DCT Coef Group:%d-%d-%d\n",activedobj.coef_arra
ys{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row2,column2),activedobj.c
oef_arrays{1,1}(row3,column3));
51.     result=result+num2str(mod(bitxor(activedobj.coef_arrays{1,1}(row1,column
1),activedobj.coef_arrays{1,1}(row3,column3),'int16'),2));
52.     fprintf("[INFO] %d\tReveal[1]:%d\t%d->%d\n",Index,activedobj.coef_arrays
{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row3,column3),mod(bitxor(act
ivedobj.coef_arrays{1,1}(row1,column1),activedobj.coef_arrays{1,1}(row3,colu
mn3),'int16'),2));
53.     result=result+num2str(mod(bitxor(activedobj.coef_arrays{1,1}(row2,column
2),activedobj.coef_arrays{1,1}(row3,column3),'int16'),2));
54.     fprintf("[INFO] %d\tReveal[2]:%d\t%d->%d\n",Index,activedobj.coef_arrays
{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,column3),mod(bitxor(act
ivedobj.coef_arrays{1,1}(row2,column2),activedobj.coef_arrays{1,1}(row3,colu
mn3),'int16'),2));
55.     %Prepare for the next Group
56.     RandTableIndex1=RandTableIndex3+1;

```

```

57.     RandTableIndex2=RandTableIndex1+1;
58.     RandTableIndex3=RandTableIndex2+1;
59.     Index=Index+1;
60.     fprintf("[SUCCESS] Revaling Success:Next Group:%d-%d-%d\t%d\n",RandTable
        Index1,RandTableIndex2,RandTableIndex3,Index);
61. end
62. plain='010000010110001101110010011011110111001101110011001000000111010001101
        0000110010100100000010001110111001001100101011000010111010000100000010101110
        1100001011011000110110000100000010101110110010100100000011000110110000101101
        1100010000001010010011001010110000101100011011010000010000001000101011101100
        1100101011100100111100100100000010000110110111101110010011011100110010101110
        0100010000001101111011001100010000001110100011010000110010100100000010101110
        1101111011100100110110001100100001000000101010100110010001100000011000100110
        1110011000100110100001110000011100000110110';%Across the Great Wall We can R
        each Every Corner of the World U201714886
63.
64. ifequal=isequal(plain,result);%Final Comparison

```