# 信息隐藏技术实验报告

Lab9 变换域隐藏法实验 网络空间安全学院 信息安全专业 2112492 刘修铭 1028

#### 1 题目

DCT域的信息隐藏包括:

- 1. 修改系数方法
- 2. 系数比较方法。

以上两种方法任选一种,实现变换域中的信息隐藏与提取。

#### 2 实验要求

写出实验报告,含程序代码和截图,word 或 pdf 格式。将实验报告、程序代码及相关文件打包压缩后(文件名命名方法: 学号-姓名-变换域隐藏法实验),提交 qq 群作业。

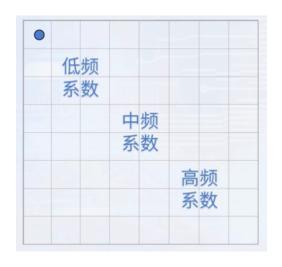
### 3 实验原理

变换域技术: 在载体的显著区域隐藏信息, 比 LSB 方法能够更好地抵抗攻击, 而且还保持了对人类感观的不可察觉性。

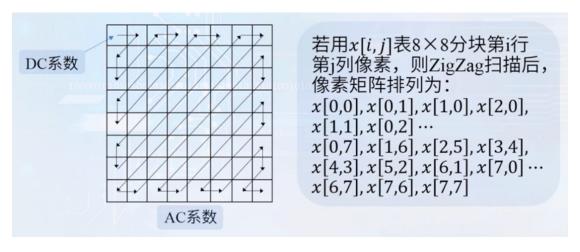
- 常用的变换域方法:
  - 离散余弦变换(DCT)
  - 离散小波变换(DWT)
  - 离散傅里叶变换(DFT)

图像压缩标准(JPEG)的核心: 二维 DCT 变换。在 DCT 域中的信息隐藏,可以有效地抵抗 JPEG 有损压缩。

二维 DCT 变换: 图像分为 8×8 的像素块,进行二维 DCT 变换,得到 8×8 的 DCT 系数。最左上角的那个洗漱为直流系数,其余为交流系数。左上角的部分为直流和低频,右下角的部分为高频,中间区域为中频。中频系数包含了图像的大部分能量,是对人的视觉最重要的部分。



DCT 与 Zig-Zag 扫描顺序:转换 2-D 系数区块成 1-D 系数,系数按照 Zig-Zag 次序排列。



基本隐藏算法: 以一定的方式挑选一些中频系数, 在这些中频系数中叠加秘密信息:

- 所有中频系数
- 固定位置的中频系数
- 随机挑选中频系数
- 选择最大的几个中频系数
- 1. 在选出的中频系数中叠加秘密信息  $x'(i,j) = x(i,j) + \alpha m_i$ , x(i,j) 为 DCT 系数, x'(i,j) 为隐藏后的 DCT 系数,  $m_i$  为第 i 个秘密信息比特,  $\alpha$  为可调参数, 控制嵌入强度。
- 2. 在选出的中频系数中叠加秘密信息  $x'(i,j) = x(i,j)(1+\alpha m_i)$ 
  - (a) 每个系数上嵌入的强度相同
  - (b) 根据系数的大小成比例地嵌入

上述的两个算法的提取都需要原始图像

3. 不需要原始载体的隐藏方法: 利用载体中两个特定数的相对大小来代表隐藏的信息

- 嵌入: 载体图像分为  $8 \times 8$  的块,做二维 DCT 变换,伪随机地选择一个图像块  $B_i$ ,分别选择其中的两个位置,比如用  $(u_1, v_1)$  和  $(u_2, v_2)$  代表所选定的两个系数的坐标。
  - 如果  $B_i(u_1,v_1) > B_i(u_2,v_2)$ , 代表隐藏 1; 如果相反,则交换两系数
  - 如果  $B_i(u_1,v_1) < B_i(u_2,v_2)$ ,代表隐藏 0; 如果相反,则交换两系数
  - 最后做二维 DCT 逆变换
- 提取:接收者进行二维 DCT 变换,比较每一块中约定位置的 DCT 系数值,根据其相对大小,得到隐藏信息的比特串,从而恢复出秘密信息。
- 特点:不需要原始图像
- 注意:如果选定位置的两个系数相差太大,则对图像影响较大,应选择相近的值(如中频系数)
- 4. 算法三的扩展: 利用 DCT 中频系数中的三个系数之间的相对关系来对秘密信息进行编码
  - 嵌入:选择三个位置 (u₁, v₁), (u₂, v₂), (u₃, v₃)
    - 嵌入 1:  $\diamondsuit B_i(u_1, v_1) > B_i(u_3, v_3) + D$ ,  $B_i(u_2, v_2) > B_i(u_3, v_3) + D$
    - 嵌入 0:  $\diamondsuit B_i(u_1, v_1) < B_i(u_3, v_3) D$ ,  $B_i(u_2, v_2) < B_i(u_3, v_3) D$

如果数据不符,就修改这三个系数值,使得它们满足上述关系。其中参数 D 的选择要考虑隐藏的健壮性和不可察觉性之间的平衡,D 越大,隐藏算法对于图像处理就越健壮,但是对图像的改动就越大,越容易引起察觉。

如果需要做的修改太大,则放弃该块,将其标识为"无效"。即,满足如下条件之一:

- $B_i(u_1, v_1) \le B_i(u_3, v_3) \le B_i(u_2, v_2)$
- $-B_i(u_2,v_2) \leq B_i(u_3,v_3) \leq B_i(u_1,v_1)$
- 提取:对图像进行 DCT 变换,比较每一块相应三个位置的系数,从它们之间的关系,可以判断隐藏的是信息 1、0 还是无效块,这样就可以恢复秘密信息。

#### 4 实验过程(含主要源代码)

本次实验选用修改系数方法进行。

- 1. Hide 函数: 将秘密图像隐藏到原始图像中
  - 将原始图像和秘密图像都分成块(大小为 8×8)
  - 对于每个块,对原始图像的块应用 DCT (离散余弦变换)
  - 根据秘密图像对应像素值,修改每个块的直流(DC)系数(DCT矩阵的左上角元素)
  - 然后对修改后的 DCT 块应用 IDCT (逆离散余弦变换)

```
1
    function result = Hide(original_img, secret_img)
 2
 3
        size = 256;
 4
        width = 8;
 5
        blocks = size / width;
 6
        result = zeros(size);
 7
 8
        for i = 1 : blocks
 9
            for j = 1 : blocks
10
                x = (i - 1) * width + 1;
11
                y = (j - 1) * width + 1;
12
13
                tmp_img = original_img(x : (x + width - 1), y : (y + width - 1));
14
                tmp_img = dct2(tmp_img);
15
16
                if secret_img(i, j) == 0
17
                    a = -1;
18
                else
19
                    a = 1;
20
                end
21
22
                tmp_img(1, 1) = (0.001 * a) + tmp_img(1, 1) * (1 + 0.001 * a);
23
                tmp_img = idct2(tmp_img);
24
                result(x : (x + width - 1), y : (y + width - 1)) = tmp_img;
25
26
            end
27
        end
28
29
    end
```

2. Extract 函数: 从包含隐藏信息的图像中提取秘密图像。将加载秘密图像后的图像(with\_secret\_img)的每个块与原始图像进行比较,并根据修改后图像的像素值是否大于原始图像的像素值,将结果图像中对应的像素设为 1 或 0。

```
function result = Extract(with_secret_img, original_img)
 2
 3
        size = 256;
 4
        width = 8;
 5
        blocks = size / width;
 6
 7
        result = ones(32);
 8
 9
        for i = 1 : blocks
10
            for j = 1 : blocks
11
                x = (i - 1) * width + 1;
12
                y = (j - 1) * width + 1;
13
14
                 if with_secret_img(x, y) > original_img(x, y)
15
                     result(i, j) = 1;
16
                 else
17
                     result(i, j) = 0;
18
                 end
```

```
19 | end 21 | end 22 | 23 | end
```

3. main 函数: 加载原始图像和秘密图像,调整大小,将秘密图像隐藏到原始图像中,然后再提取出来

```
1
    function main()
 2
 3
        clc;
 4
        clear all;
 5
        close all;
 6
 7
        original_img = (imread('./pic/icon_gray.bmp'));
 8
        secret_img = imbinarize(imread('./pic/star_gray.bmp'));
 9
10
        figure();
11
        original_img = imresize(original_img, [256, 256]);
12
         imshow(original_img);
13
         imwrite(original_img, './pic/DCT/original.bmp');
14
15
        figure();
16
         secret_img = imresize(~secret_img, [32,32]);
17
         imshow(secret_img);
18
         imwrite(secret_img, './pic/DCT/reverse_secret.bmp');
19
20
        original_img = double(original_img) / 256;
21
         secret_img = im2double(secret_img);
22
23
        figure();
24
        with_secret_img = Hide(original_img, secret_img);
25
         imshow(with_secret_img);
26
         imwrite(with_secret_img, './pic/DCT/with_secret.bmp');
27
28
        figure();
29
        extract_img = Extract(with_secret_img, original_img);
30
         imshow(extract_img);
31
         imwrite(extract_img, './pic/DCT/extract.bmp');
32
33
    end
```

### 5 实验结果及分析

如图是原始灰度图像。



接着是原始的灰度秘密图像。



对秘密图像的二值图像取反并进行形状处理,得到如下图像。



调用函数,对秘密图像进行隐藏,得到如下图像。



然后提取秘密图像,得到如下结果。



可以看到,与前面的图像基本相同,说明实验成功。

# 6 参考

本次实验主要参考慕课完成。

# 7 说明

本次实验所有代码均已放在 codes 文件夹下。

 $1 \mid \mathtt{dct.m}$ 

本次实验所有图片均位于 codes/pic 文件夹