南开大学
Nankai University

南　　开　　大　　学

网 络 空 间 安 全 学 院

信息隐藏技术实验九

变换域隐藏法实验

穆禹宸　2012026

年级：2020 级

专业：信息安全、法学双学位班

指导教师：李朝晖

2023 年 4 月 25 日

摘要

利用 MATLAB 对变换域隐藏的算法进行实践。将图像进行分块，逐块修改某些提前选定的位置 DCT 系数来来隐藏秘密信息的比特。

关键字：二值图像，信息隐藏，变换域隐藏

目录

一、 实验要求

(一) 实验目的

DCT 域的信息隐藏包括:

1. 修改系数方法
2. 系数比较方法

利用其中一种方法事先变换域的信息隐藏和提取。

(二) 实验环境

所需实验环境

主要有如下内容

(1) 运行系统: Windows11

(2) 实验工具: Matlab2022a

(3) 数据: BMP 格式图像

(三) 实验要求

1. 在 Matlab 之中完成
2. 编写实验代码和报告，并给出截图
3. QQ 群提交作业

二、 实验原理

(一) 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

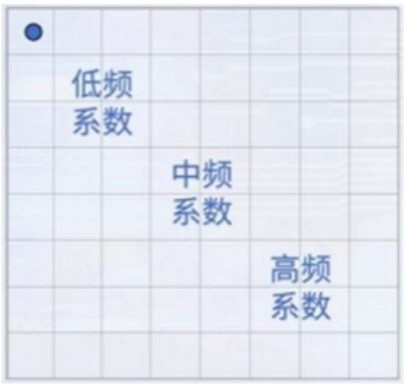

图 1: 二维 DCT 变换

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 8x8 的像素块，进行二维 DCT 变换，得到 8x8 的 DCT 系数。最左上角的系数是直流系数其余是交流系数。左上角部分是直流和低频，右下角部分是高频，中间区域是中频。中低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。

(二) 信息隐藏算法

以一定方式选择一些中频系数，在这些中频系数中叠加秘密信息：

算法一：在选出的中频系数中叠加秘密信息 x(i, j)' = x(i, j) + a mi

算法二：在选出的中频系数中叠加秘密信息 x(i, j)' = x(i, j) (1 + a mi)

算法三：不需要原始载体，直接利用载体中两个特定数的相对大小来代表隐藏的信息。

算法四：算法三的扩展，利用 DCT 中频系数中的三个特定系数的相对关系来对秘密信息进行编码。

【注】如果选定位置的两个系数的下观察太大，则对图像的影响较大。应选择相近的值 (如中频系数)。对图像进行 DCT 变换，利用每一块特定位置的系数关系或系数大小中判断隐藏的信息是“1”“0”还是无效块，这样就可以恢复出秘密信息。

(三) 修改系数法

一般而言隐藏算法可以是修改系数的方法，也可以是系数比较的方法，为了方便起见我们实现前者。DCT 域修改的系数一般都是中频系数，我们可以在这些中频系数中叠加所需的秘密信息。其算法二原理为：在选中的中频系数中叠加秘密信息，成比例修改 DCT 系数。

$$x'(i,j) = x(i,j) + a \ldots m_i, \alpha \in (0,1)$$

其中，a 是可调参数，控制嵌入强度。与算法一相比，每个系数上嵌入的强度大小会有所不同，和原始系数比例相关。

其中 $x'(i,j)$ 和 $x(i,j)$ 分别为 DCT 系数和隐藏后的系数，α 是一个超参数用以控制嵌入的强度，m_i 就是第 i 个信息比特。

实验将通过修改图片的方式嵌入秘密信息，然后将新的图片与原图进行比较来提取秘密信息。此外所有的图片都应该是位图。

三、 实验步骤

实验步骤

主要有如下三个步骤:

(1) 将用于当做载体的图片大小设置为 256*256

(2) 将隐藏的图片设置为 64*64，同时将它取反

(3) 然后将图像转换为 double 送入 DCT 算法中进行处理

1. 实验代码

秘密信息嵌入

```
%% CalculateBlack
clc;clear all;close all;
img = (imread('./raw.bmp'));
watermark = imbinarize(imread('./watermark.bmp'));
img = imresize(img, [256, 256]);
watermark = imresize(~watermark, [64,64]);
```

读取原图像文件数据 Lena 图像和水印图像 NK 图像并调整图像大小和位深度主要是对图像进行处理。

以下是部分使用到的图片:

图 2: NK

第一行代码 img = (imread('./raw.bmp')); 是读取名为"raw.bmp" 的图像文件，并将其存储在变量 img 中。

第二行代码 watermark = imbinarize(imread('./watermark.bmp')); 是读取名为"watermark.bmp" 的图像文件, 并将其转换为二值图像, 即只有黑白两种颜色的图像, 并将其存储在变量 watermark 中。

第三行代码 img = imresize(img, [256, 256]); 是将变量 img 中的图像大小调整为 256x256 像素，即将图像进行缩放。

第四行代码 watermark = imresize(watermark, [64,64]); 是将变量 watermark 中的图像进行反转（黑变白，白变黑），并将其大小调整为 64x64 像素，即将水印图像进行缩放。

这段代码的作用是将原始图像和水印图像进行处理，以便后续的图像水印嵌入操作。

随后是划分块 block 并在而每一个块的某一个特定位置修改 DCT 系数，嵌入秘密信息。

嵌入水印，逐块进行扫描。

```
img = double(img)/256;
watermark = im2double(watermark);
size = 256; width = 4;

blocks = size / width;
new_image = zeros(size);
vec = ones(64);

for i = 1 : blocks
    for j = 1 : blocks
        x = (i - 1) * width + 1;
        y = (j - 1) * width + 1;
        cur = img(x:x+width-1, y:y+width-1);
        cur = dct2(cur);

        if watermark(i, j) == 0
            a = -1;
        else
            a = 1;
        end

        cur(1, 1) = cur(1, 1) * (1 + .01 * a) + .01 * a;
        cur = idct2(cur);
        new_image(x: x + width - 1, y : y + width - 1) = cur;
    end
end

```

dct2() 是二维离散余弦变换;idct2() 是二维离散余弦逆变换函数。主要是对图像进行嵌入水印操作。

第一行代码 img = double(img)/256; 是将变量 img 中的图像转换为 double 类型，并将其值除以 256，以便后续的计算。

第二行代码 watermark = im2double(watermark); 是将变量 watermark 中的图像转换为 double 类型，并将其值归一化到 0 到 1 之间。

第三行代码 size = 256; width = 4; 是定义了图像的大小为 256x256 像素，每个小块的大小为 4x4 像素。

第四行代码 blocks = size / width; 是计算出图像中小块的数量。

第五行代码 new_image = zeros(size); 是创建一个大小为 256x256 的全零矩阵，用于存储嵌入水印后的图像。

接下来的代码使用了两个 for 循环，分别遍历了图像中的每个小块，并对其进行处理。具体来说，对于每个小块，首先使用 DCT 变换将其转换为频域，然后根据水印图像的值，对小块的第一个系数进行微调，以嵌入水印信息。最后，使用 IDCT 变换将小块转换回空域，并将其存储

到新的图像中。

最后一行代码 subplot(234); imshow(new_image,[]);title(" 嵌入水印"); 是将嵌入水印后的图像显示出来，并在图像上方添加一个标题" 嵌入水印"。

然后，秘密信息提取。

```
for i = 1 : blocks
    for j = 1 : blocks
        x = (i - 1) * width + 1;
        y = (j - 1) * width + 1;

        if new_image(x, y) > img(x, y)
            vec(i, j) = 1;
        else
            vec(i, j) = 0;
        end
    end
end
```

提取水印：逐块进行比较。主要是对图像进行水印提取操作。

第一行代码 img = double(img)/256; 是将变量 img 中的图像转换为 double 类型，并将其值除以 256，以便后续的计算。

第二行代码 watermark = im2double(watermark); 是将变量 watermark 中的图像转换为 double 类型，并将其值归一化到 0 到 1 之间。

第三行代码 size = 256; width = 4; 是定义了图像的大小为 256x256 像素，每个小块的大小为 4x4 像素。

第四行代码 blocks = size / width; 是计算出图像中小块的数量。

第五行代码 new_image = zeros(size); 是创建一个大小为 256x256 的全零矩阵，用于存储提取出的水印信息。

第六行代码 vec = ones(64); 是创建一个大小为 64x64 的全 1 矩阵，用于存储每个小块中的水印信息。

接下来的代码使用了两个 for 循环，分别遍历了图像中的每个小块，并对其进行处理。具体来说，对于每个小块，首先比较其在原始图像中的第一个像素值和在提取出的图像中的第一个像素值的大小关系，如果前者大于后者，则将 vec(i,j) 赋值为 1，否则赋值为 0。

最后一行代码 end 是结束了 for 循环的语句块。

2. 实验结果

实验的结果如下：

图 3: 实验结果

可以观察到，嵌入以后的图像与原图像几乎没有变化，只有色调深度上有细微差距，进行原水印图像和提取出的水印图像对比，发现完全一致，提取成功!

同时，我对比了不同图片和不同大小的图像的结果，如下图所示:

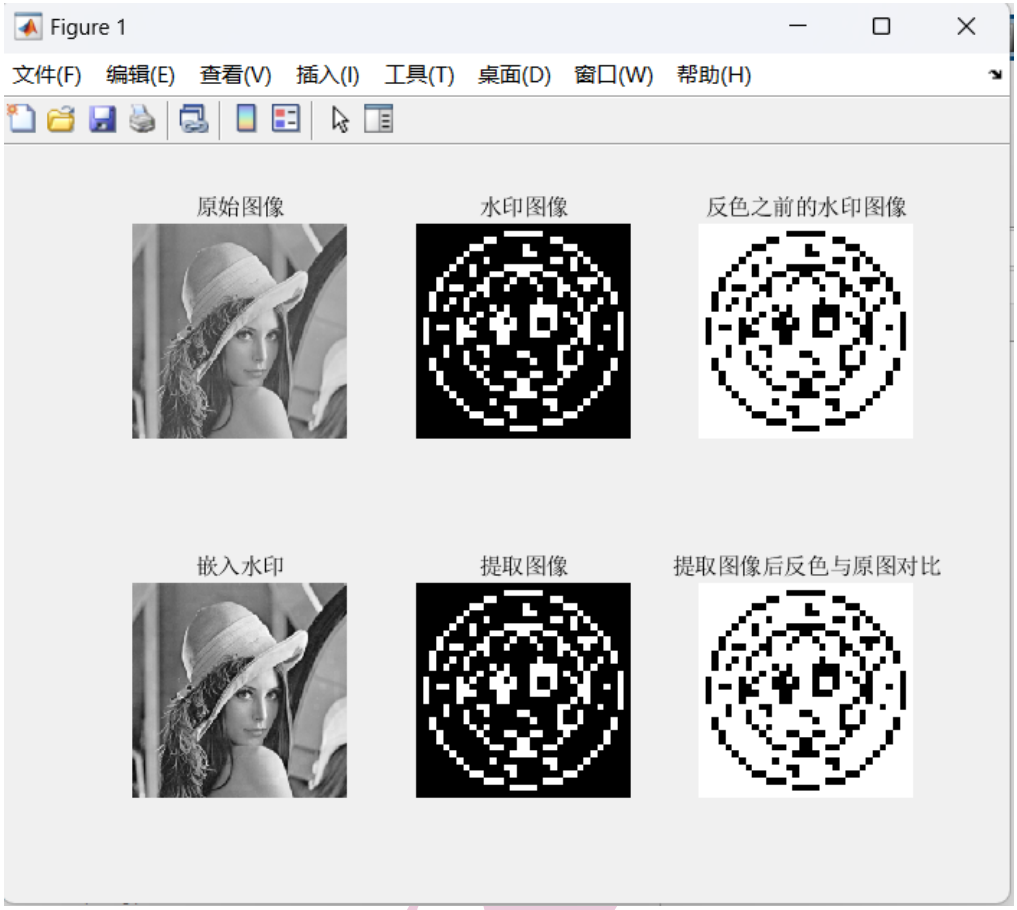

图 4: 32*32 实验结果

我发现如果图片大小为 32*32，那么会使原图非常模糊，丢失大量信息。
这里是其他图片的结果展示：


Figure 1
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
水印图像
反色之前的水印图像
嵌入水印
提取图像
提取图像后反色与原图对比

图 5: 功夫熊猫实验结果

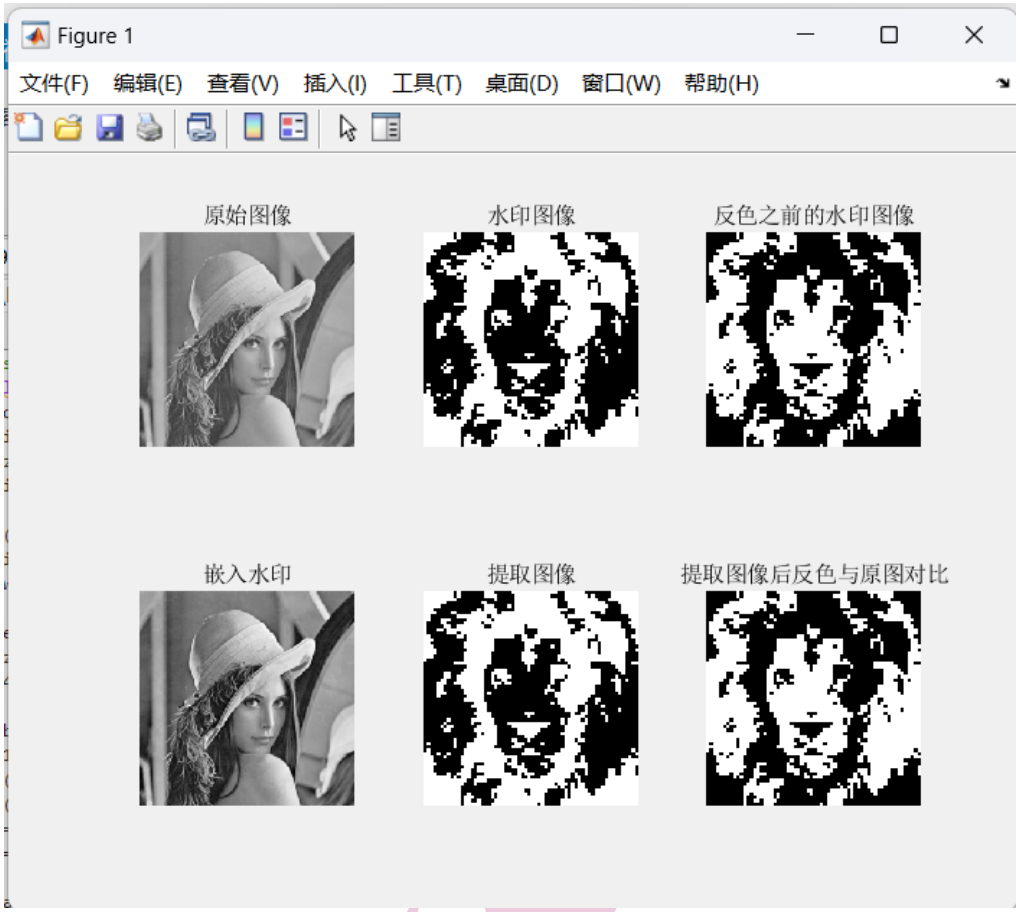

图 6: 狮子实验结果

可见实验取得圆满成功!

四、 实验心得体会

在这个实验中，我们学习了一种图像水印技术——变换域隐藏法。这种技术可以将水印信息嵌入到图像的 DCT 变换系数中，从而实现对图像的保护和认证。在实验中，我们使用 MATLAB 语言编写了相关的代码，并对图像进行了嵌入水印和提取水印的操作。通过这个实验，我深刻地认识到了图像水印技术的重要性和应用价值。

首先，我认为图像水印技术是一种非常重要的信息安全技术。在现代社会中，图像已经成为人们生活中不可或缺的一部分，而图像水印技术可以有效地保护图像的版权和隐私。通过将水印信息嵌入到图像中，可以防止他人对图像进行盗用和篡改，从而保护图像的合法权益。此外，图像水印技术还可以用于图像的认证和溯源，可以帮助人们追踪图像的来源和使用情况，从而保障图像的安全和可信度。

其次，我认为图像水印技术是一种非常有应用价值的技术。在现代数字媒体领域中，图像水印技术已经得到了广泛的应用。例如，在数字版权保护方面，图像水印技术可以用于保护数字图书、音乐、视频等数字媒体的版权；在数字证据保全方面，图像水印技术可以用于保护数字证据的完整性和可信度；在数字图像处理方面，图像水印技术可以用于图像的去噪、增强和压缩等方面。因此，图像水印技术具有非常广泛的应用前景和市场潜力。

最后，我认为通过这个实验，我不仅学习了图像水印技术的基本原理和实现方法，还学习了如何使用 MATLAB 语言进行图像处理和编程。这对于我今后的学习和工作都具有非常重要的意

义。通过这个实验，我深刻地认识到了图像水印技术的重要性和应用价值，也更加熟练地掌握了MATLAB 语言的使用方法。我相信这些知识和技能对于我的未来发展和职业规划都将产生积极的影响。

总之，这个实验让我受益匪浅，不仅扩展了我的知识面，还提高了我的实践能力和编程技能。我相信这些经验和体会将对我的未来学习和工作产生积极的影响，也希望能够在今后的学习和工作中继续深入研究图像水印技术，为保护数字媒体的安全和可信度做出自己的贡献。

五、 附录：完整实验代码

完整实验代码

```
% Pre-process images.
clc;clear all;close all;
img = (imread('./raw.bmp'));
watermark = imbinarize(imread('./watermark.bmp'));
img = imresize(img, [256, 256]);
watermark = imresize(~watermark, [64,64]);

img = double(img)/256;
watermark = im2double(watermark);
size = 256; width = 4;

blocks = size / width;
new_image = zeros(size);
vec = ones(64);

for i = 1 : blocks
    for j = 1 : blocks
        x = (i - 1) * width + 1;
        y = (j - 1) * width + 1;
        cur = img(x:x+width-1, y:y+width-1);
        cur = dct2(cur);

        if watermark(i, j) == 0
            a = -1;
        else
            a = 1;
        end

        cur(1, 1) = cur(1, 1) * (1 + .01 * a) + .01 * a;
        cur = idct2(cur);
        new_image(x: x + width - 1, y : y + width - 1) = cur;
    end
end

for i = 1 : blocks
    for j = 1 : blocks
        x = (i - 1) * width + 1;
        y = (j - 1) * width + 1;

        if new_image(x, y) > img(x, y)
            vec(i, j) = 1;
        else
            vec(i, j) = 0;
        end
```

```
    end
end

subplot(231); imshow(img);title('原始图像');
subplot(232); imshow(watermark);title('水印图像');
subplot(233); imshow(imcomplement(watermark));title('反色之前的水印图像');
subplot(234); imshow(new_image,[]);title('嵌入水印');
subplot(235); imshow(vec,[]);title('提取图像');
subplot(236); imshow(imcomplement(vec),[]);title('提取图像后反色与原图对比');
```