# 图像信息隐藏算法一

钮心忻、杨榆、雷敏
北京邮电大学 计算机学院 信息安全中心
yangyu@bupt.edu.cn

# 图像信息隐藏技术分类

- 图像隐写算法
  - 隐蔽性
- 图像鲁棒性水印
  - 能够抵抗各种信号处理、攻击
- 图像脆弱性水印
  - 完整性验证、篡改定位

# 课程内容

1. 格式信息隐藏算法
2. 时域算法
3. 变换域算法
4. 扩频算法
5. 统计算法

# 文件格式信息隐藏

- 图像文件都有一定的存贮格式
- 文件头主要描述图像文件的格式、文件大小、数据起始偏移地址、图像数据大小等关键信息
- 利用图像文件的这种特性，可以在图像文件中隐藏秘密数据

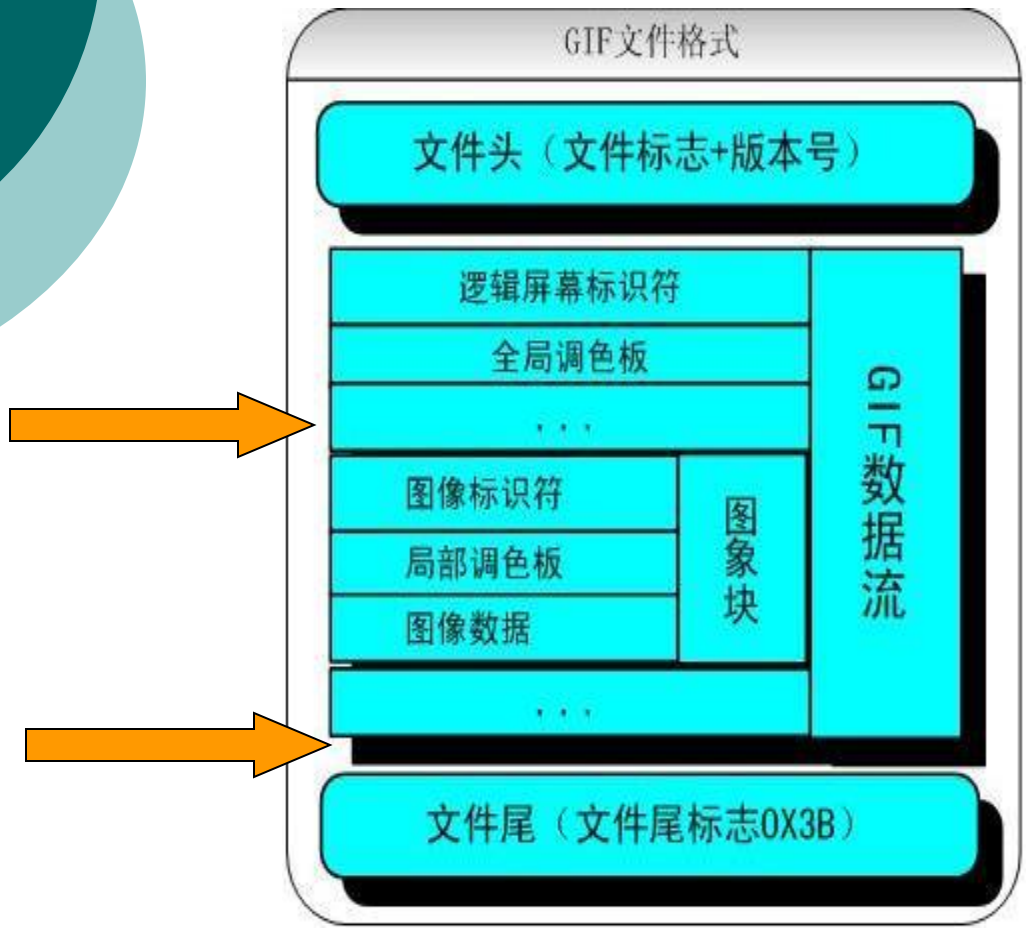
# BMP格式


文件头部：54字节

文件头信息包含BMP文件的类型、图像数据偏移地址、大小和打印格式等信息;

位图头信息包含BMP文件的尺寸定义等信息

# 文件格式信息隐藏




1。秘密信息嵌入文件注释块

2。秘密信息嵌入文件末尾

DataStash等掩密软件

# BMP图像文件

- 秘密数据必须保存到位图文件的两个有效数据结构之间
  - 位图图像的结尾
  - 文件头数据与图像数据, …
- 隐藏在文件尾部需修改文件头中文件长度域
- 隐藏在文件头与图像数据之间，需要修改文件长度、数据起始偏移地址

# 文件格式信息隐藏

- 例

| 比较 | 隐写前 | 隐写后 |
| --- | --- | --- |
| 外观（无变化） | | |
| 文件大小 | 263,222字节 | 483,382字节 |
| 文件头 | 文件长度域值为0X40436 | 文件长度域值为0X76036 |
| 文件尾 | 图像数据后文件结束 | 图像数据后增加Office文档数据 |

# 文件格式信息隐藏

- 演示：
  - 内容：命令行实现的格式隐藏。
  - 方法：利用dos命令copy将两幅图像合并成为一幅图像。
  - 分析：
    - 按二进制流形式合并载体和水印文件为一体，
    - 预览图片时，位于合成图像起始位置的载体图像的显示信息生效，因此合成图像看来与载体图像完全相同，但实际合成图像包含了水印图像。

# 在HTML文件中隐藏信息

- HTML文件是文本文件，在浏览器端仅能显示ASCII码中的可见字符。
- 利用这一特点，可以在HTML的标记之间插入隐藏的数据。
- 比如，如果要隐藏的二进制比特值为1，在选定的HTML标记后插入ASCII码值为9的字符；
- 如果要隐藏的二进制比特值为0，则在选定的HTML标记后插入ASCII码值为32的字符 。

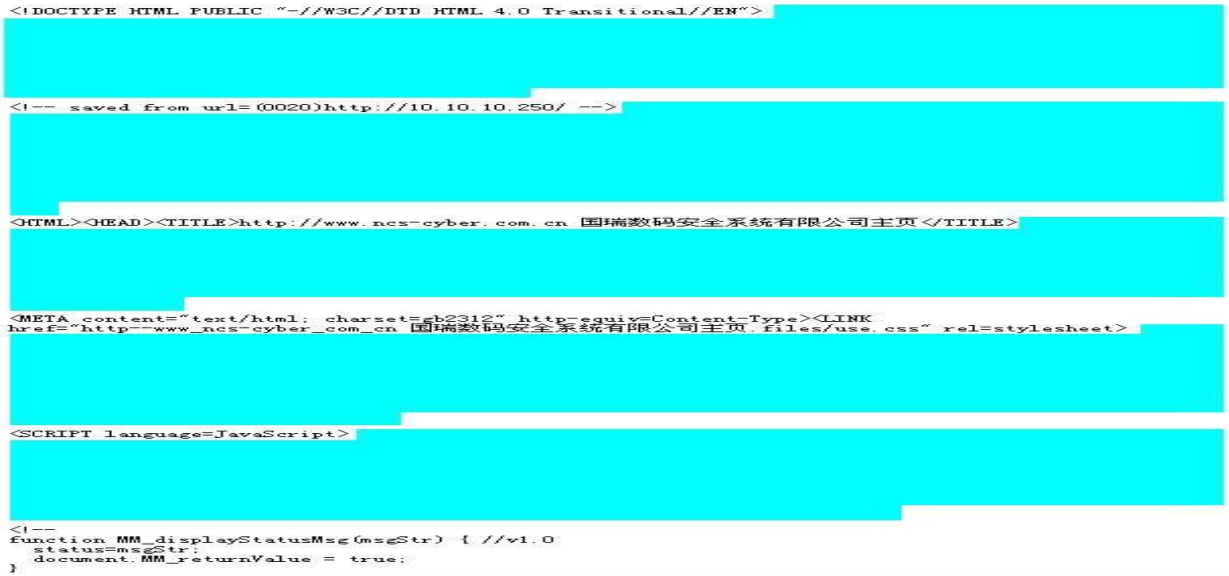
# 在HTML文件中隐藏信息：示例

- 隐藏信息前的HTML文件的部分文本

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<!-- saved from url=(0020)http://10.10.10.250/ -->
<HTML><HEAD><TITLE>http://www.ncs-cyber.com.cn 国瑞数码安全系统有限公司主页</TITLE>
<META content="text/html; charset=gb2312" http-equiv=Content-Type><LINK
href="http--www_ncs-cyber_com_cn 国瑞数码安全系统有限公司主页.files/use.css" rel=stylesheet>
<SCRIPT language=JavaScript>
<!--
function MM_displayStatusMsg(msgStr) { //v1.0
  status=msgStr;
  document.MM_returnValue = true;
}
```

# 在HTML文件中隐藏信息：示例

- 隐藏信息后的HTML文件的相应文本

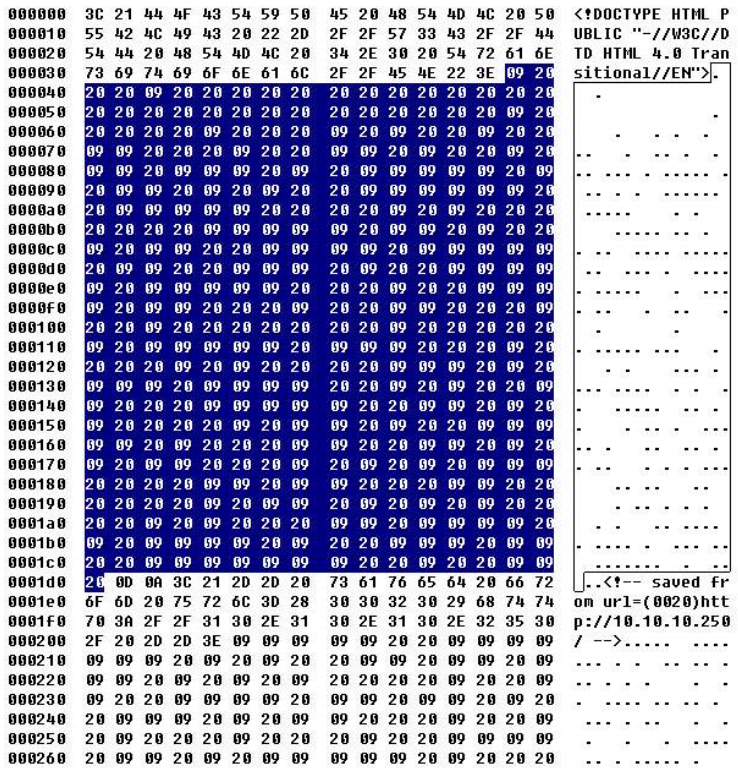


```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<!-- saved from url=(0020)http://10.10.10.250/ -->

<HTML><HEAD><TITLE>http://www.ncs-cyber.com.cn 国瑞数码安全系统有限公司主页</TITLE>

<META content="text/html; charset=gb2312" http-equiv=Content-Type><LINK
href="http--www_ncs-cyber_com_cn 国瑞数码安全系统有限公司主页_files/use.css" rel=stylesheet>

<SCRIPT language=JavaScript>

<!--
function MM_displayStatusMsg(msgStr) { //v1.0
  status=msgStr;
  document.MM_returnValue = true;
}
```

- 深颜色部分为隐藏的数据。尽管在代码文本上可以明显的看出二者之间的差异，但在浏览器端则显示不出任何差异

# 在HTML文件中隐藏信息：示例

- 在HTML文件中隐藏了秘密数据的载体文件的部分数据



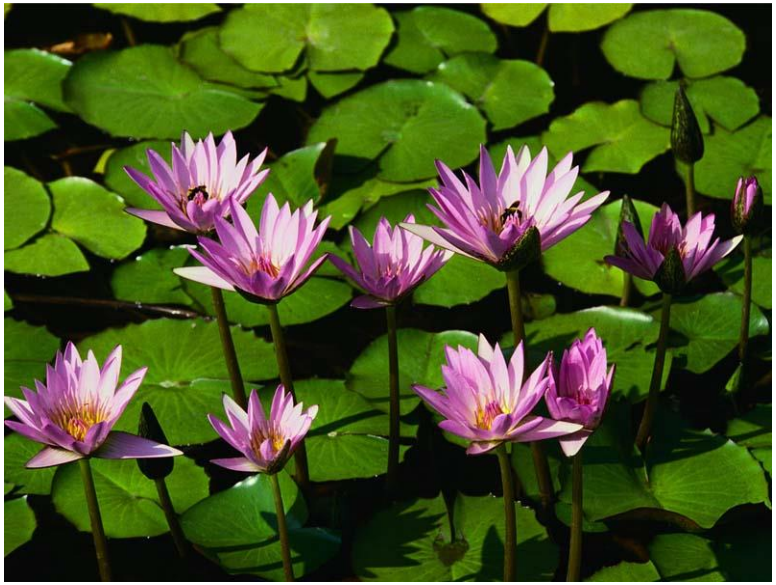
# 在HTML文件中隐藏信息：示例

（a）隐藏数据之前的浏览器显示

（b）隐藏数据之后的浏览器显示

# 在HTML文件中隐藏信息：示例

- HTML文件中隐藏的秘密数据



# 文件格式信息隐藏——小结

- 容量分析
  - 平均每样点可以嵌入隐藏多少比特信息?
  - 嵌入信息于文件格式中，嵌入量与像素个数没有直接关系。
  - 因此，文件中可以隐藏“任意多”的数据。

# 文件格式信息隐藏——小结

- 透明性分析
  - 嵌入过程不修改像素，不会对载体本身的外观造成任何损坏。
- 稳健性分析
  - 文件的拷贝不会对隐藏的信息造成破坏。
  - 文件存取工具在保存文档时可能会造成隐藏数据的丢失。

# 文件格式信息隐藏——小结

- 安全性分析
  - 自然图像文件所需存储空间可根据图像分辨率和文件类型估算，
  - 嵌入较多信息使隐写图像大小远远超出正常值，隐藏的信息较容易被发现。
  - 为了确保隐藏内容的机密性，需要首先进行加密处理，然后再隐藏。

# 课程内容

1. 格式信息隐藏算法
2. 时域算法
3. 变换域算法
4. 扩频算法
5. 统计算法

# 2.0 时域替换技术

- 任何多媒体信息，在数字化时，都会产生物理随机噪声，而人的感官系统对这些随机噪声是不敏感的
- 替换技术就是利用这个原理，试图用秘密信息比特替换掉随机噪声，以达到隐藏秘密信息的目的

# 2.0 时域替换技术

- 图像的位平面概念

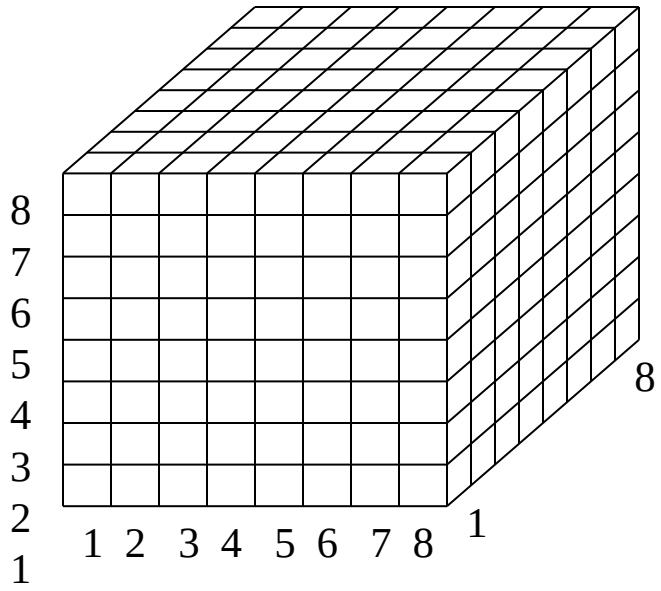


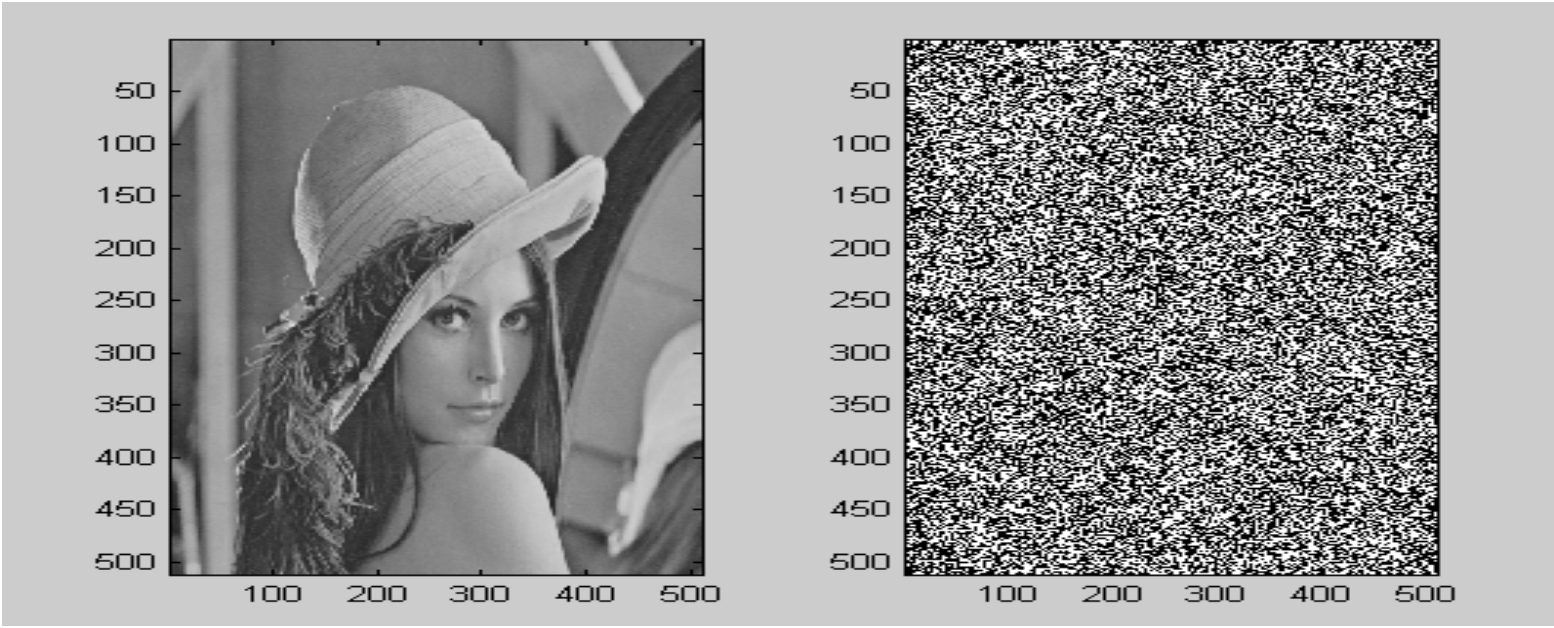

图4－1，图像象素的灰度表示

# 2.0 时域替换技术

- 图像各个位平面的作用
  - 原始图像（8bit 灰度BMP图像）

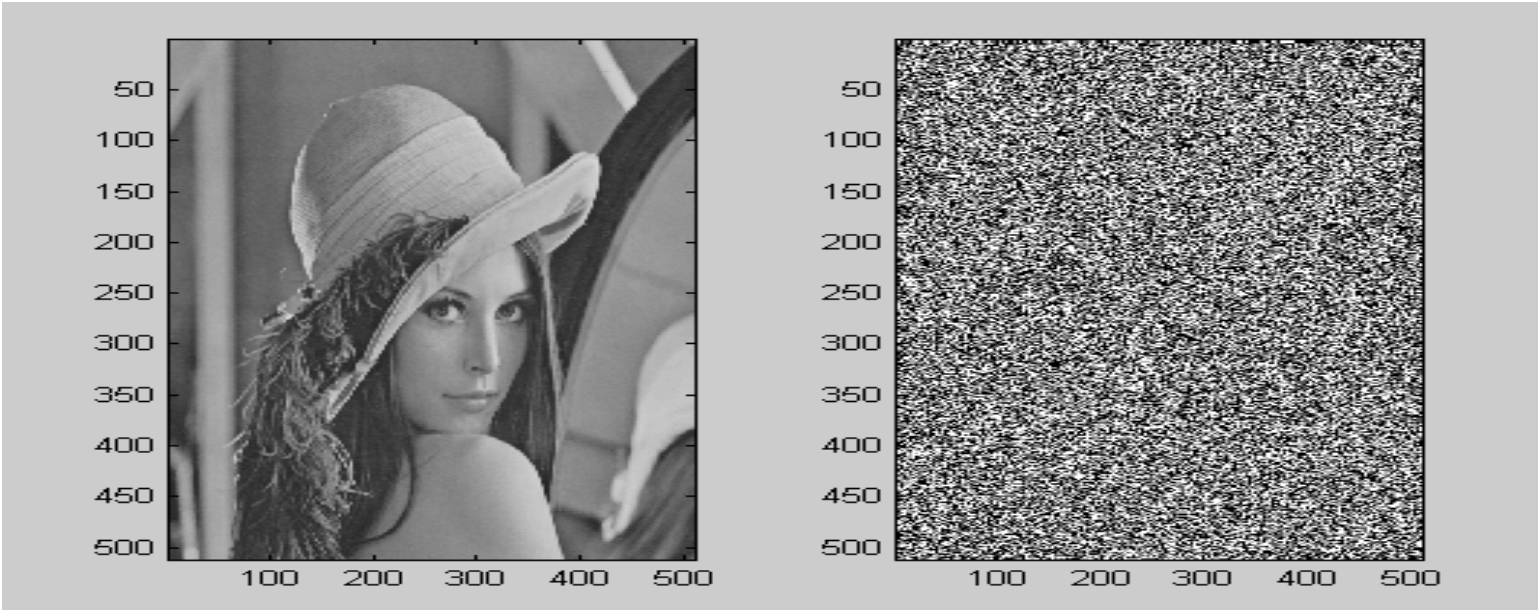
# 2.0 时域替换技术

- 图像各个位平面的作用
  - 去掉第一个位平面的Lena图像和第一个位平面



# 2.0 时域替换技术

- 图像各个位平面的作用
  - 去掉第1－2个位平面的Lena图像和第1－2个位平面



# 2.0 时域替换技术

LSB（Least Significant Bit)算法

图像的低位权比特平面，即，由像素较低有效位组成的平面，对视觉影响较小，可将其替换为秘密信息。$y_i = x_i \& 0\text{xFE} + m_i; x_i, y_i \in \{0,1,2,\cdots 255\}, m_i \in \{0,1\}$

- 接收方提取相同的比特平面就可以获取秘密信息。$\tilde{m}_i = \tilde{y}_i \& 0\text{x}01; \tilde{y}_i \in \{0,1,2,\cdots 255\}, \tilde{m}_i \in \{0,1\}$
- 为增强安全性，收发双方也可协商秘钥。发方使用密钥将像素位置置乱后，再隐藏信息。$y_{k_i} = x_{k_i} \& 0\text{xFE} + m_i; k_i \in \{0,1,2,\cdots \text{M} \times \text{N}\}, x_{k_i}, y_{k_i} \in \{0,1,2,\cdots 255\}, m_i \in \{0,1\}$, M和N分别是图像的行数和列数。

# 2.0时域替换技术


LSB
提升容量
MLSB，TLSB
提升安全性
±1，±k
细粒度
BPCS
变换域
EzStego，Jstego…
自适应
EA，S-UNIWARD…

# 2.0 时域替换技术

- LSB改进算法
  - 除最低比特平面外，图像其它低比特平面也可隐藏信息。替换多个图像比特平面隐藏信息的算法，称为MLSB（Multiple Least Significant Bit）算法。
    - 把比特平面作为整体，用秘密信息替换各个比特平面，这种算法称为IMLSB(Independent MLSB)
    - 把各像素低*n*比特作为整体，依次用秘密信息替换，这种算法称为TMLSB(Truss MLSB)

# 2.0 时域替换技术

- IMLSB算法案例
  - 已知某灰度图像，像素值用4比特表示，局部区域像素矩阵如下：
  - $x[i,j]=\begin{bmatrix}9 & 11\\ 12 & 13\end{bmatrix}_D$
  - 请问：
    - 若要隐藏字符‘0’（ASCII码0x30），应该如何修改这些像素？（要求待隐藏的信息按列优先排列，先替换最低比特平面，再替换次低比特平面）
    - 若约定像素位置扰乱方式为转置，要求先扰乱，再隐藏，则收发双方应该如何处理？

# 2.0 时域替换技术

- IMLSB算法案例
  - 解：无扰乱方式处理步骤为：
  - 第一：将字符转为比特串，并按像素矩阵尺寸排列：
    - ’0’->X(30)->B(0011 0000)-〉按列优先限排列-〉
    - $b_0[i,j]=\begin{bmatrix}0 & 1\\0 & 1\end{bmatrix}, b_1[i,j]=\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
  - 第二：按位权低高展开像素矩阵
    - $x[i,j]=\begin{bmatrix}9 & 11\\12 & 13\end{bmatrix}_D=\begin{bmatrix}1001 & 1011\\1100 & 1101\end{bmatrix}\rightarrow x_0[i,j]=$ $\begin{bmatrix}1 & 1\\0 & 1\end{bmatrix}, x_1[i,j]=\begin{bmatrix}0 & 1\\0 & 0\end{bmatrix}, x_2[i,j]=\begin{bmatrix}0 & 0\\1 & 1\end{bmatrix}, x_3[i,j]=\begin{bmatrix}1 & 1\\1 & 1\end{bmatrix}$

# 2.0 时域替换技术

- IMLSB算法案例
  - 解：无扰乱方式处理步骤为：
  - 第三：用$b_0$替换$x_0$，$b_1$替换$x_1$，得到：
    - $x_0[i,j]=\begin{bmatrix}0 & 1\\ 0 & 1\end{bmatrix}, x_1[i,j]=\begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix}, x_2[i,j]=\begin{bmatrix}0 & 0\\ 1 & 1\end{bmatrix}, x_3[i,j]=\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix}$
    - 即隐藏前，像素为：$\begin{bmatrix}9 & 11\\ 12 & 13\end{bmatrix}_D$，隐藏后：

$$\mathrm{x}[i,j]=\begin{bmatrix}1000 & 1001\\ 1100 & 1101\end{bmatrix}=\begin{bmatrix}8 & 9\\ 12 & 13\end{bmatrix}_D$$

# 2.0 时域替换技术

- IMLSB算法案例
  - 解：无扰乱方式处理步骤为：
  - 第四：接收方拿到含秘密信息的图像块后：
    - $x[i,j]=\begin{bmatrix}8 & 9\\12 & 13\end{bmatrix}_D=\begin{bmatrix}1000 & 1001\\1100 & 1101\end{bmatrix}$
  - 提取最低和次低比特平面：
    - $b_0[i,j]=\begin{bmatrix}0 & 1\\0 & 1\end{bmatrix}, b_1[i,j]=\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$
  - 按列优先排列，得到：
    - B(00110000)->X(30)->’0’

# 2.0 时域替换技术

- IMLSB算法案例
  - 解：扰乱方式为转置时，处理步骤为：
  - 第一：仍然将字符转为比特串，并按像素矩阵尺寸排列：
    - ’0’->X(30)->B(0011 0000)-〉 按列优先限排列-〉
    - $b_0[i,j]=\begin{bmatrix}0 & 1\\ 0 & 1\end{bmatrix}, b_1[i,j]=\begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix}$

# 2.0 时域替换技术

- IMLSB算法案例
  - 解：扰乱方式为转置时，处理步骤为：
  - 第二：嵌入前，先扰乱像素得到：
    - $x^T[i,j] = \begin{bmatrix} 9 & 11 \\ 12 & 13 \end{bmatrix}^T = \begin{bmatrix} 9 & 12 \\ 11 & 13 \end{bmatrix}_D$
  - 再按位权低高展开像素矩阵(9,11,12,13)
    - $x^T[i,j] = \begin{bmatrix} 9 & 12 \\ 11 & 13 \end{bmatrix}_D = \begin{bmatrix} 1001 & 1100 \\ 1011 & 1101 \end{bmatrix} -> x_0[i,j] = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, x_1[i,j] = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, x_2[i,j] = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}, x_3[i,j] = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

# 2.0 时域替换技术

- IMLSB算法案例
  - 解：扰乱方式为转置时，处理步骤为：
  - 第三：用$b_0$替换$x_0$，$b_1$替换$x_1$，得到：
    - $x_0[i,j]=\begin{bmatrix}0 & 1\\ 0 & 1\end{bmatrix}, x_1[i,j]=\begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix}, x_2[i,j]=\begin{bmatrix}0 & 1\\ 0 & 1\end{bmatrix}, x_3[i,j]=\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix}$
    - 即隐藏后：$x^T[i,j]=\begin{bmatrix}1000 & 1101\\ 1000 & 1101\end{bmatrix}=\begin{bmatrix}8 & 13\\ 8 & 13\end{bmatrix}_D$
    - 恢复原始排列顺序：
    - $x[i,j]=\begin{bmatrix}8 & 8\\ 13 & 13\end{bmatrix}_D$

# 2.0 时域替换技术

- IMLSB算法案例
  - 解：扰乱方式为转置时，处理步骤为：
  - 第四：接收方拿到含秘密信息的图像块后，首先按约定好的方使扰乱图像块：
    - $x^T[i,j] = \begin{bmatrix} 8 & 8 \\ 13 & 13 \end{bmatrix}^T_D = \begin{bmatrix} 8 & 13 \\ 8 & 13 \end{bmatrix}_D = \begin{bmatrix} 1000 & 1101 \\ 1000 & 1101 \end{bmatrix}$
  - 提取最低和次低比特平面：
    - $\mathrm{b}_0[i,j] = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}, b_1[i,j] = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$
  - 按列优先排列，得到：
    - B(00110000)->X(30)->’0’

# 2.0 时域替换技术

- LSB改进算法
  - LSB算法通过替换像素比特为秘密信息实现隐藏，会影响载体的统计特性。
  - 改进思路是，修改像素时，无论像素值是奇数还是偶数，都可以对其加1或减1来实现隐藏。基于该思路的算法称为±1隐藏。

# 2.0 时域替换技术

- LSB改进算法
  - 一种±1隐藏算法
  - 设秘密信息位为$w$，对应隐藏该位的像素灰度值为$x(i, j)$
    - 如果$w$ 与$x(i, j)$的最低比特位相同，即$w = x(i, j) \bmod 2$，那么不改变原始数据
    - 当$w$ 与$x(i, j)$的最低比特位不同，即$w \neq x(i, j) \bmod 2$ 时，计算

$$T = \sum_{u=i-1}^{i+1} \sum_{v=j-1}^{j+1} x(u, v) - 9 \cdot x(i, j)$$

# 2.0 时域替换技术

- LSB改进算法
  - 一种±1隐藏算法
    - 对$x(i,j)$作如下调整

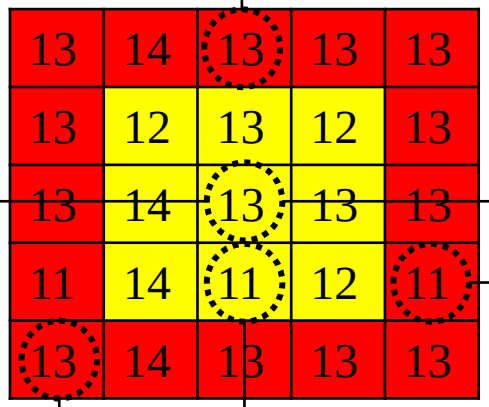
$$x(i,j)=\begin{cases} x(i,j)-1 & , \quad \text{if } T \le 0, 0 < x(i,j) < 255 \\ x(i,j)+1 & , \quad \text{if } T > 0, 0 < x(i,j) < 255 \\ x(i,j)-1 & , \quad \text{if } x(i,j)=255 \\ x(i,j)+1 & , \quad \text{if } x(i,j)=0 \end{cases}$$

# 2.0 时域替换技术

- LSB改进算法
  - 一种±1隐藏算法

| | | |
|---|---|---|
| 12 | 13 | 12 |
| 14 | 13 | 13 |
| 14 | 11 | 12 |

| | | |
|---|---|---|
| 0 | 1 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |




$x_{i,j}$垂直方向扩展元素$x_{i-1,j}$镜像自$x_{i+1,j}$

$x_{i,j}$对角线方向扩展元素$x_{i+1,j-1}$镜像自$x_{i-1,j+1}$

$x_{i,j}$水平方向扩展元素$x_{i,j+1}$镜像自$x_{i,j-1}$

中心像素周围8邻域像素均值为13，中心像素值为12，小于平均水平，隐藏后值+1

| | | |
|---|---|---|
| 12 | 13 | 13 |
| 14 | 13 | 13 |
| 13 | 12 | 12 |

中心像素周围8邻域像素均值为13.25，中心像素值为11，小于平均水平，隐藏后值+1

中心像素周围8邻域像素均值为13，中心像素值为14，大于平均水平，隐藏后值-1

# 2.0 时域替换技术

- BPCS（bit-plane complexity segmentation）位平面复杂度分割
- 原理:
  - 将载体数据的多个位平面分成小块
  - 人的视觉对变化剧烈、复杂度较高的位平面小块不敏感
  - 秘密信息可以加载在多个位平面

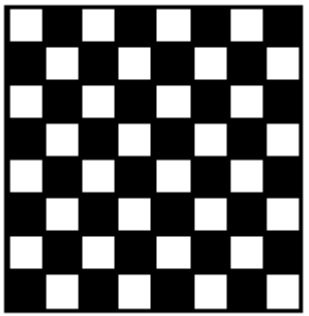
| 样点 | |
|---|---|
| 3 | 2 |
| 1 | 0 |

| 第2比特平面 | |
|---|---|
| 1 | 1 |
| 0 | 0 |

| 第1比特平面 | |
|---|---|
| 1 | 0 |
| 1 | 0 |

# 2.0 时域替换技术

- BPCS算法:复杂度
  - 复杂度定义：所有相邻像素对中取值不同的像素对数目。
  - 复杂度最大可能值记为Cmax
    - 例如，8×8，复杂度0—112
    - 全0或全1：复杂度0
    - 0、1相间棋盘状：复杂度112



  - 复杂度为C的图像块，与棋盘状图像块做异或运算后，生成的新图像块，复杂度为：Cmax-C
  - 图像块与棋盘状图像块的异或操作，称之为共轭处理。

# 2.0 时域替换技术

- BPCS算法信息嵌入步骤：
  - 将载体图像所有位平面分为小块，如8×8；
  - 计算每个小块的复杂度；
  - 选择复杂度大于αCmax的位平面小块用于负载秘密信息；
  - 将秘密信息组成位平面小块，如果其复杂度大于αCmax，则直接替换原位平面小块；
  - 如果其复杂度小于等于αCmax，则先作共轭处理，再替换；
  - 记录下哪些小块经过共轭处理。

# 2.0 时域替换技术

- BPCS算法信息提取步骤：
  - 将载体图像所有位平面分为小块，如8×8；
  - 计算每个小块的复杂度；
  - 选择所有复杂度大于aCmax的位平面小块取出，可得到含共轭处理的秘密信息；
  - 根据共轭处理记录，对于作过共轭处理的小块，再作一次共轭，就可以得到秘密信息。

# 2.0 时域替换技术

- BPCS案例：
  - 设分块大小为2*2,每像素值用3 比特表示，alpha=0.4。
  - 1、Cmax为多少?
  - 2、若已知小块像素（行优先排列）为（7,5,4,6），其各个比特平面的复杂度为?
  - 3、有哪些比特平面适于隐藏信息?

# 2.0 时域替换技术

- BPCS案例
- 解：
  - 1、Cmax=4,alpha = 0.4,
  - alpha * Cmax = 0.4 * 4 = 1.6

# 2.0 时域替换技术

- BPCS案例
- 解：2~3、
- 像素为7,5 即： 111 101
- 4,6 100 110
- 三个比特平面，按位权低高分别为：
- （1 1） （1 0） （1 1）
- （0 0） （0 1） （1 1）
- 其复杂度为2，4,0，因此可以隐藏秘密信息的是最低和次低比特平面

# 2.0 时域替换技术

- BPCS案例：
  - 设秘密信息为
  - 1 1 和 0 0
  - 1 0　 0 0
  - 则隐藏秘密信息后，上述像素值变为？约定先替换位权较低的比特平面。
  - 约定棋盘小块为：
  - 1 0
  - 0 1

# 2.0 时域替换技术

- BPCS案例，解：
- 则第一块1 1复杂度为2，大于alpha*Cmax
- 1 0
- 可以直接替换，
- 第二块要0 0复杂度为0，小于alpha*Cmax
- 0 0
- 应与棋盘小块1 0异或，得：1 0
- 0 1 0 1

# 2.0 时域替换技术

- BPCS案例： 解：
- 替换后，三个比特平面，从低到高分别为
- （1 1） （1 0） （1 1）
- （1 0） （0 1） （1 1）
- 还原为十进制值为：
- 7 5
- 5 6

略过深入讨论

# 2.0 时域替换技术

- BPCS深入讨论
  - 关于阈值$\alpha C_{max}$
    - BPCS算法约定:
      - 只从复杂度大于$\alpha C_{max}$的位平面小块中提取信息;
      - 嵌入时，若秘密信息小块复杂度小于$\alpha C_{max}$，则作共轭;
      - 由以上两点可知：$C_{max} - \alpha C_{max} > \alpha C_{max}$
      - 所以：$2\alpha C_{max} < C_{max}$，即$\alpha < 0.5$

# 2.0 时域替换技术

- BPCS深入讨论
  - 一般不采用二进制形式划分位平面，而是采用循环码划分位平面
  - 二进制划分位平面：会有许多小块复杂度大于0.5Cmax
  - 循环码划分位平面：绝大多数小块复杂度小于0.5Cmax

# 2.0 时域替换技术

- BPCS深入讨论
  - 二进制码和循环码的互换
  - B：二进制
  - G：循环码

$$G_{N-1} = B_{N-1}, G_{N-2} = B_{N-2} \oplus B_{N-1}, G_{N-3} = B_{N-3} \oplus B_{N-2}, \cdots, B_0 = B_0 \oplus B_1$$

$$B_{N-1} = G_{N-1}, B_{N-2} = G_{N-2} \oplus B_{N-1}, B_{N-3} = G_{N-3} \oplus B_{N-2}, \cdots, B_0 = G_0 \oplus B_1$$

# 2.0 时域替换技术

- BPCS深入讨论

表 6.1.1　数字 0～7 的二进制码和循环码

| 十进制 | 二进制 | 循环码 |
|---|---|---|
| 0 | 000 | 000 |
| 1 | 001 | 001 |
| 2 | 010 | 011 |
| 3 | 011 | 010 |
| 4 | 100 | 110 |
| 5 | 101 | 111 |
| 6 | 110 | 101 |
| 7 | 111 | 100 |

# 2.0 时域替换技术

- BPCS深入讨论——案例：
  - 设分块大小为2*2,每像素值用3 比特表示，alpha=0.4。像素使用循环码编码。若要在小块 $x[i,j]=\begin{bmatrix}7 & 5\\ 4 & 6\end{bmatrix}_D$ 中隐藏$b_0[i,j]=\begin{bmatrix}1 & 1\\ 1 & 0\end{bmatrix}$，$b_1[i,j]=\begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix}$（约定优先替换位权较低的比特平面），则收发双方应如何处理?

# 2.0 时域替换技术

- BPCS深入讨论——案例：
  - 解：
  - 1.先对x$[i,j]$进行循环编码：
  - 由$G_{N-1}=B_{N-1},G_{N-2}=B_{N-1}\oplus B_{N-2},\cdots G_0=B_1\oplus B_0,\oplus$为按位异或运算符，可知：$7_D=111$编码为:$G_2=B_2=1,G_1=B_2\oplus B_1=1\oplus 1=0,G_0=B_1\oplus B_0=1\oplus 1=0$,依此类推:
  - x$[i,j]$=$\begin{bmatrix}7 & 5\\ 4 & 6\end{bmatrix}_D=\begin{bmatrix}111 & 101\\ 100 & 110\end{bmatrix}$,编码得:
    $\begin{bmatrix}100 & 111\\ 110 & 101\end{bmatrix}$

# 2.0 时域替换技术

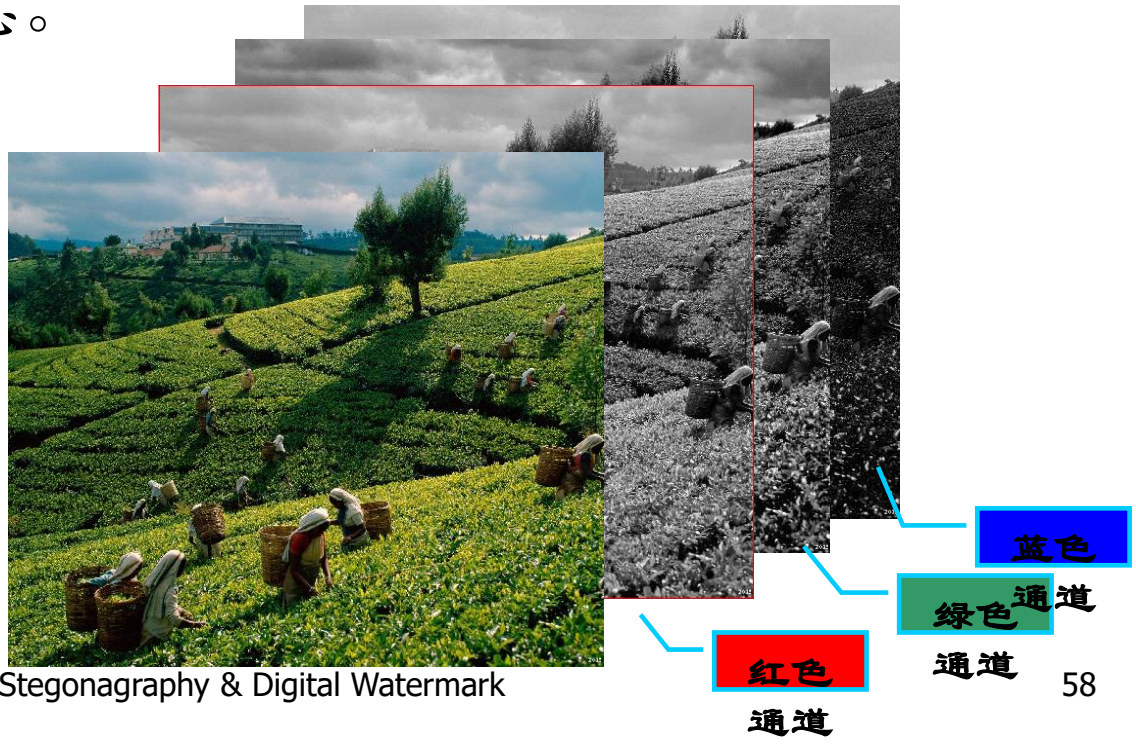
- BPCS深入讨论——案例：
  - 解：
  - 2.对循环编码后$x[i,j]$进行隐藏：
  - $G[i,j]=\begin{bmatrix}100 & 111\\ 110 & 101\end{bmatrix}$, $G_0[i,j]=\begin{bmatrix}0 & 1\\ 0 & 1\end{bmatrix}$,复杂度$C_0=2$
  - $G_1[i,j]=\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$,复杂度$C_1=4$，$G_2[i,j]=\begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix}$,复杂度$C_2=0$，$G_0, G_1$可以隐藏。
  - 后续步骤参考普通BPCS算法处理，请独立完成。

# 2.1 基于颜色索引顺序的隐藏算法

- 调色板的图像
  - 调色板数据：定义了N种颜色索引对（$i,c_i$）
  - 图像数据：代表每一个像素的调色板索引
  - 颜色向量代表R、G、B三个分量的值，如果是灰度图像，则三个分量取值相同
- 使用调色板格式，可以降低图像存储的大小

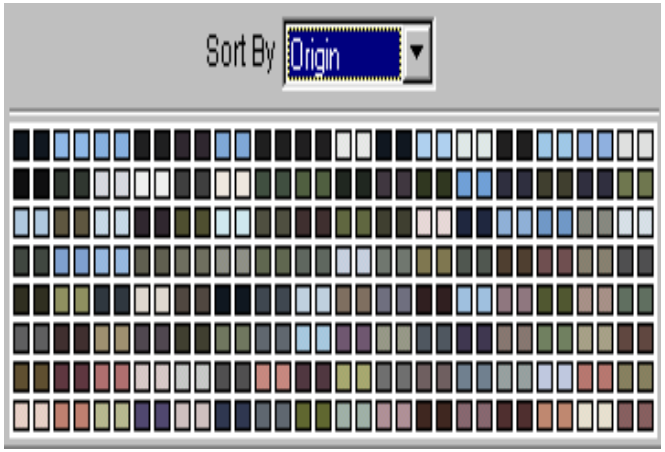
# 2.0 时域替换技术

- LSB算法
  - LSB算法可应用于彩色图像。典型方法为：依次在彩色图像的各个颜色通道，使用LSB算法嵌入信息。



# 2.1 基于颜色索引顺序的隐藏算法

- Hide&Seek将调色板中各颜色分量进行划分，生成一个特殊的调色板，然后提取128个基色并经过细微的修改后扩展到256个颜色，产生可互相替换的相近颜色，嵌入时根据嵌入数据进行相应的替换
  - 产生相近颜色对



# 2.1 基于颜色索引顺序的隐藏算法

- 调色板设计激发的思路
  - 设调色板有红、绿、蓝三种颜色，分别给三种颜色分配索引编号，有几种分配方法?

| 方案编号 | 索引0 | 索引1 | 索引2 |
|---|---|---|---|
| 0 | 红 | 绿 | 蓝 |
| 1 | 红 | 蓝 | 绿 |
| 2 | 蓝 | 红 | 绿 |
| 3 | 蓝 | 绿 | 红 |
| 4 | 绿 | 蓝 | 红 |
| 5 | 绿 | 红 | 蓝 |

# 2.1 基于颜色索引顺序的隐藏算法

- 调色板设计激发的思路
  - 共有六种方案，任意一种皆可行。
  - 能否利用方案的选择传递秘密信息?
  - 例如
    - 选择方案0，表示发送的信息00
    - 选择方案1，表示发送的信息01。。。
  - 思路是否可行取决于能否建立起颜色方案（颜色排列顺序）和整型的映射关系。

# 2.1 基于颜色索引顺序的隐藏算法

- 算法思路
  - 关联秘密信息与颜色排列方式，实现：
    - 已知秘密信息便能确定颜色排列，
    - 已知颜色排列便能确定秘密信息。
  - 也就是说，寻找一对函数,使得:
    - $y = g(x)$和$x = g^{-1}(y)$
    - 其中$x = (x_1, x_2, x_3 \cdots x_m)$，表示M比特秘密信息
    - $y = (c_{t1}, c_{t2}, c_{t3} \cdots c_{tn})$，表示N钟颜色的一个排列

# 2.1 基于颜色索引顺序的隐写

- 算法容量分析
  - 设图像可嵌入M比特秘密信息，有N种颜色，
  - 则
    - M比特秘密信息能表示 $2^M$ 个整数
    - 颜色有N！种排列方法
    - 要保证 $\left[0,1,2\cdots,2^M-1\right]$ 中每个整数都有对应的颜色排列，则应满足下列关系

      $$2^M \le N!$$

    - 所以可以嵌入 $\left\lfloor \log_2(N!) \right\rfloor$ 比特秘密信息。（$\lfloor . \rfloor$,floor，向下取整）

# 2.1 基于颜色索引顺序的隐写

- 排列到整数的映射——排列的符号表示
  - N个颜色 $c_0, c_1 \cdots, c_{N-1}$ 的任一排列可用 $\left(c_{t(0)} c_{t(1)} \cdots c_{t(N-1)}\right)$ 表示。
  - 其中 $t(i) = 0,1,\cdots N-1$
  - 例如，若定义 $t(i) = N-1-i$
  - 则颜色排列方案为 $(c_{N-1} c_{N-2} \cdots c_0)$
  - 例如， N为4时，
    - 对于排列 $(c_0, c_1, c_2, c_3)$， $t(i) = i$
    - 对于排列 $(c_2, c_1, c_0, c_3)$， $t(0) = 2, t(1) = 1, t(2) = 0, t(3) = 3$

# 2.1 基于颜色索引顺序的隐写

- 排列到整数的映射——逆序函数定义
  - 选定一种排列方案为标准排列。
    - 例如，选定$(c_0c_1c_2\cdots c_{N-1})$为颜色$c_0,c_1\cdots,c_{N-1}$的标准排列。
  - 对于任意排列，$(c_{t(0)}c_{t(1)}\cdots c_{t(N-1)})$，定义$f(i)$为这个排列中，在$c_{t(i)}$之前且在标准排序中应该在$c_{t(i)}$之后的元素个数。
    - 例如，对于排列$(c_{N-1}c_{N-2}\cdots c_0)$，$f(0)=0$，$f(1)=1$
    - $f(2)=2$，$f(i)=i$，$f(N-1)=N-1$
    - 例如，N为3，标准排列为$(c_0,c_1,c_2)$时，对于排列$(c_2,c_0,c_1)$, $f(0)=0, f(1)=1, f(2)=1$

# 2.1 基于颜色索引顺序的隐写

- 排列到整数的映射——排列函数定义
  - 定义排列函数 $s=\sum_{i=0}^{N-1} f(i)i!$
    - 例如，对于排列 $(c_{N-1}c_{N-2}\cdots c_0)$

$$s=f(0)0!+f(1)1!+\cdots+f(N-1)(N-1)!$$
$$=1\times1+2\times2!+\cdots+(N-1)\times(N-1)!$$

    - 例如，N=3，标准排列为$(c_0,c_1,c_2)$时，
      - 对于排列$(c_2,c_0,c_1)$,
      - $s=f(0)\times0!+f(1)\times1!+f(2)\times2!=0\times1+1\times1+1\times2=3$
  - 使用排列函数，可将一种排列方案唯一地映射为一个整数。

# 2.1 基于颜色索引顺序的隐写

- 整数到排列的映射——逆排列函数定义
  - 给定任意整数 $s \le 2^M - 1$，其中 $M = \lfloor \log_2(N!) \rfloor$
  - 则可以唯一确定一个排列
  - 根据 $f(i) = \left\lfloor \dfrac{s - \sum_{j=i+1}^{N-1} f(j)\, j!}{i!} \right\rfloor$ ，
  - 依次计算 $f(N-1), f(N-2), \cdots, f(0)$，可确定排列
  - 例如：$f(N-1) = \left\lfloor \dfrac{s - \sum_{j=N}^{N-1} f(j)\, j!}{(N-1)!} \right\rfloor = \left\lfloor \dfrac{s}{(N-1)!} \right\rfloor$

# 2.1 基于颜色索引顺序的隐藏算法

- 例：
  - 有4种颜色，标准排列为 $(c_0c_1c_2c_3)$
  - 则哪一种排列的对应的函数值为13?
  - 解：
  - 先求：

$$f(3)=\left\lfloor \frac{13-\sum_{j=4}^{3} f(j)\,j!}{3!} \right\rfloor = \left\lfloor \frac{13-0}{6} \right\rfloor = 2$$

  - 当前排列第4个元素前有三个元素，其中两个元素在标准排列中应在该元素之后。

# 2.1 基于颜色索引顺序的隐藏算法

- 例：
  - 因此，第四个元素应该是 $c_1$，即排列为：

$$\left(c_{t(0)}c_{t(1)}c_{t(2)}c_1\right)$$

  - 以此类推：

$$f(2)=\left\lfloor\frac{13-\sum_{j=3}^{3}f(j)\,j!}{2!}\right\rfloor=\left\lfloor\frac{13-2\times3!}{4}\right\rfloor=\left\lfloor\frac{1}{4}\right\rfloor=0$$

  - 当前排列第3个元素前有2个元素，0个元素在标准排列中应在该元素之后。

# 2.1 基于颜色索引顺序的隐藏算法

- 例：
  - 因此，第3个元素应该是 $c_3$ ，即排列为：

$$\left(c_{t(0)}c_{t(1)}c_3c_1\right)$$

  - 以此类推：
  - f(1)=1，当前排列第2个元素前有1个元素，而它在标准排列中应在该元素之后，即为 $c_0$
  - f(0)=0，即为 $c_2$
  - 所以排列为：

$$\left(c_2c_0c_3c_1\right)$$

# 2.1 基于颜色索引顺序的隐藏算法

隐藏流程

1、对图像中出现的真实颜色根据一定规律排序（使用密钥产生三元组，根据真实颜色与三元组的距离排序），并记录序号为 $c_o, c_1, c_2 \ldots c_{N-1}$。N 种颜色的排列数为 N!。可以隐藏 m=$\lfloor \log_2 N! \rfloor$ 比特秘密信息。

2、提取 m 比特秘密信息，根据映射函数，使得 $(m)_d = s\left(c_{t(0)}, c_{t(1)}, c_{t(2)} \ldots c_{t(N-1)}\right)$，即信息的十进制值与颜色的某种排列对应。

3、替换调色板的色对的索引编号，以及每个像素的索引。

# 2.1 基于颜色索引顺序的隐藏算法

解隐藏流程

1、 获取三元组，对接收图像真实颜色排序

2、 根据调色板索引获取排序

3、 根据 $(m)_d = s\left(c_{t(0)}, c_{t(1)}, c_{t(2)} \ldots c_{t(N-1)}\right)$ 计算， 提取秘密信息

# 2.1 基于颜色索引顺序的隐藏算法

- 例
  - 设调色板仅有4种颜色，分别为（167、142、172）、（162、175、210）、（214、167、172）、（176、205、231）
  - 则可通过下列方法产生标准排序:
  - 由密钥产生的三元组为（135.6，202.7，82.3），则根据d=sqrt((R0-R)^2+(G0-G)^2+(B0-B)^2)距离排序可得标准排序。

# 2.1 基于颜色索引顺序的隐藏算法

例

| 序号 | 颜色值 | | | 与（135.6，202.7，82.3）的欧氏距离 |
|---|---|---|---|---|
| | R | G | B | |
| C0 | 162 | 175 | 210 | 133.3 |
| C1 | 176 | 205 | 201 | 125.4 |
| C2 | 214 | 167 | 172 | 124.4 |
| C3 | 167 | 146 | 172 | 110.7 |

# 2.1 基于颜色索引顺序的隐藏算法

- 例
  - 2、四种颜色可隐藏4比特秘密信息，若要隐藏的秘密信息为0101，
  - 则由(0101)d=5，根据公式，依次计算f(N-1),…,f(1)，可以得到颜色排序应为(C2,C1,C0,C3)。
  - 也可通过查表法，反查排序，得对应序列(C2,C1,C0,C3)

# 2.1 基于颜色索引顺序的隐藏算法

例，即此时调色板为：

| 序号 | 颜色值 | | | 与（135.6，202.7，82.3）的欧氏距离 |
|---|---|---|---|---|
| | R | G | B | |
| 0(C2) | 214 | 167 | 172 | 124.4 |
| 1(C1) | 176 | 205 | 201 | 125.4 |
| 2(C0) | 162 | 175 | 210 | 133.3 |
| 3(C3) | 167 | 146 | 172 | 110.7 |

# 2.1 基于颜色索引顺序的隐藏算法

- 例
  - 3、若接收方发现颜色排列为（C3,C1,C0,C2）
  - 则由f(0)=0,f(1)=1,f(2)=2,f(3)=1可知:
  - 排列函数值为s=0*1+1*1+2*2!+1*3!=11
  - 即其中秘密信息比特为：1011

# 2.1 基于颜色索引顺序的隐藏算法

- 小结
  - 优点：
    - 不改变图像的显示效果
  - 缺点：
    - 嵌入量与图像颜色数有关，不随图像尺寸变化。
    - 图像处理软件会根据亮度、出现频率重排调色板，会删除秘密信息。

# 2.2 思考

- 可以采用LSB修改颜色向量以隐藏数据吗?
- 可以采用LSB对调色板图像像素进行处理，以隐藏数据吗?
- 如果可以，算法可能有什么优缺点，有什么改进方法?

# 2.3 基于调色板图像内容的隐藏

EzStego

1、 将调色板颜色按照亮度依次排序， $Y = 0.299R + 0.587G + 0.114B$

2、 为每个颜色分配一个亮度序号

3、 使用密钥，对应秘密比特和像素

4、 根据秘密比特对像素颜色的亮度进行 LSB 密写

亮度序号

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 7 | 4 | 1 | 3 | 5 | 0 | 6 | 2 |
| 7,4 | 7,4 | 1,3 | 1,3 | 5,0 | 5,0 | 6,2 | 6,2 |

颜色序号

优点：简单，容易实现

缺点：尽管隐藏后，像素的亮度变化不大，但颜色可能有明显变化。例如（95,0,0）和(0,0,250)亮度接近，前者淡红，后者深蓝

# 2.3 基于调色板图像内容的隐藏

- 例一
  - 已知颜色亮度可通如下过近似公式
    - Y=0.3*R+0.6*G+0.1B
  - 且已知某图像调色板为：（亮度按升序排列）
    - 0：<24，231，117>（浅绿）(Y0=157.5)
    - 1：<40，215，206>（青）(Y1=161.6)
    - 2：<251，241，57>（明黄）(Y2=225.6)
    - 3：<238，70，87>（桃红）(Y3=122.1)
  - 问1：若在值为013231的像素上使用EzStego隐藏比特“010101”，则像素值变为？
  - 问2：若已知图像经过EzStego处理，且像素值为013231，则可提取秘密信息比特为？

# 2.3 基于调色板图像内容的隐藏

- 例一
  - 解，由近似公式计算得Y0~=158, Y1~=162, Y2~=226, Y3~=122，
  - 颜色亮度序号为：1，2，3，0
  - 像素值（颜色索引）为013231，对应亮度编号为120302，
  - 在亮度编号上用LSB嵌入010101，则亮度号变为：030303，
  - 因此，隐写后，像素值变为：323232

# 2.3 基于调色板图像内容的隐藏

- 例一
  - 解，若隐写后像素值（颜色索引）为013231，
  - 则其亮度编号为120302，
  - 已知隐藏算法为亮度域上LSB，
  - 因此秘密信息为：100100

# 最佳奇偶分配隐写

- 计算不同颜色c(k)、c(l)之间的距离d(k,l)
- 将所有d(k,l)按从小到大排序
- 集合C初始值为空
- 选择一个d(k,l)，要求c(k)，c(l)至少有一个不属于C
  - 若有多个同时满足条件，随机选择其中之一
  - 若不存在满足条件的d(k,l)，退出循环
  - 若c(k)、c(l)皆不属于C，令P(k)=0,P(l)=1，将c(k)、c(l)加入C
  - 若c(k)属于C，c(l)不属于C，令P(l)=1-P(k)，将c(l)加入C
  - 若c(k)不属于C，c(l)属于C，令P(k)=1-P(l)，将c(k)加入C

# 最佳奇偶分配隐写

| c(1) c(2) c(3) c(4) | | |
|---|---|---|
| d(1,2) | d(1,3) | d(1,4) |
| | d(2,3) | d(2,4) |
| | | d(3,4) |

| C | {} |
|---|---|
| P(1) | |
| P(2) | |
| P(3) | |
| P(4) | |

d(1,2)

| C | {c(1),c(2)} |
|---|---|
| P(1) | 0 |
| P(2) | 1 |
| P(3) | |
| P(4) | |

d(1,3)

| C | {c(1),c(2),c(3)} |
|---|---|
| P(1) | 0 |
| P(2) | 1 |
| P(3) | 1 |
| P(4) | |

d(1,4)

| C | {c(1),c(2),c(3),c(4)} |
|---|---|
| P(1) | 0 |
| P(2) | 1 |
| P(3) | 1 |
| P(4) | 1 |

# 最佳奇偶分配隐写

- 最佳奇偶分配方法
  - 将调色板中的颜色划分为两个子集，分别代表0，1比特
  - 与某一颜色距离最小的另一颜色属于不同的子集
- 特点：失真小

# 2.4 二值图像中的信息隐藏

- 二值图像
  - 由黑白像素的分布构成的图像
  - 例如：传真、文字识别等
- 常用信息隐藏方法
  - 利用图像区域中黑色像素的个数隐藏秘密信息

# 2.4 二值图像中的信息隐藏

- 方法一
  - 基于黑白像素比例的隐藏
- 嵌入
  - 划分一个二值图像为$L(m)$ 个矩形图像区域$B_i$
  - 调整黑白像素的比例，使其满足：隐藏0时，黑色像素比例大于50%，隐藏1时则相反
  - 修改应遵循不引起感观察觉的原则
  - 修改应在黑白区域的边缘进行

# 2.4 二值图像中的信息隐藏

- 应考虑有一定的冗余度，确定有效区域
- 确定两个阈值 $R_1 > 50\%$ 和 $R_0 < 50\%$ 以及一个稳健性参数 $\lambda$。隐藏1时，该块的黑色像素的个数应属于 $[R_1, R_1 + \lambda]$；隐藏0时，该块的黑色像素的个数应属于 $[R_0 - \lambda, R_0]$
- 如果为了适应所嵌入的比特，目标块必须修改太多的像素，就把该块设为无效
- 标识无效块：将无效块中的像素进行少量的修改，使得其中黑色像素的百分比大于 $R_1 + 3\lambda$ 或者小于 $R_0 - 3\lambda$

# 2.4 二值图像中的信息隐藏

- 提取
  - 判断每一个图像块黑色像素的百分比，如果大于$R_1+3\lambda$，或者小于$R_0-3\lambda$，则跳过这样的无效块
  - 如果在$[R_1, R_1+\lambda]$或者 $[R_0-\lambda, R_0]$的范围内，则正确提取出秘密信息0或1

# 2.4 二值图像中的信息隐藏

- 方法二
  - 基于游程编码的隐藏
- 游程
  - 连续的黑色或白色像素称为一个游程
  - 用游程的起始位置和游程长度编码
- 例
  - 如下像素可用游程编码为


  - $< a_0 ,3>,< a_1,5>< a_2,4>< a_3,2>< a_4,1>$

# 2.4 二值图像中的信息隐藏

- 嵌入
  - 修改二值图像的游程长度
  - 隐藏0：修改该游程长度为偶数
  - 隐藏1：修改游程长度为奇数
  - 若秘密信息的取值与游程长度的奇偶性相匹配，则不改变游程长度
- 提取
  - 根据游程长度的奇偶性提取出秘密信息

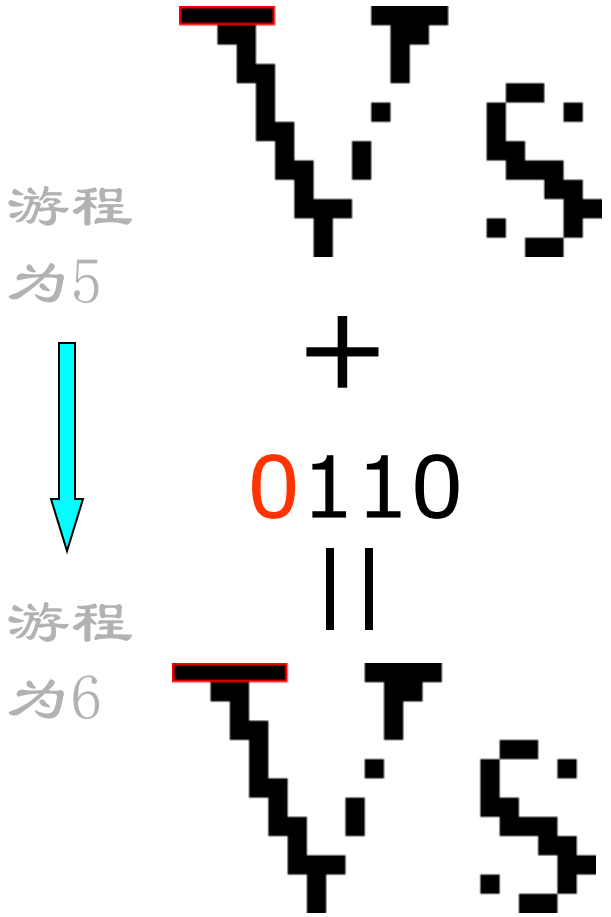
# 2.4 二值图像中的信息隐藏

- 例如：
  - 游程为： $<a_0,3>,<a_1,5><a_2,4><a_3,2><a_4,1>$
  - 若依次嵌入比特“0”、“1”，则
    - 游程$<a_0,3>$的长度为3，嵌入比特“0”，长度变为2，游程变为$<a_0,2>$，其后一个游程相应改变为$<a_1-1,6>$，所以完整游程为： $<a_0,2>,<a_1-1,6><a_2,4><a_3,2><a_4,1>$
    - 在游程$<a_1-1,6>$嵌入比特“1”，长度变为7，游程变为$<a_1-1,7>$，后一个游程变为$<a_2+1,3>$，所以完整游程为： $<a_0,2>,<a_1-1,7><a_2+1,3><a_3,2><a_4,1>$

# 2.4 二值图像中的信息隐藏

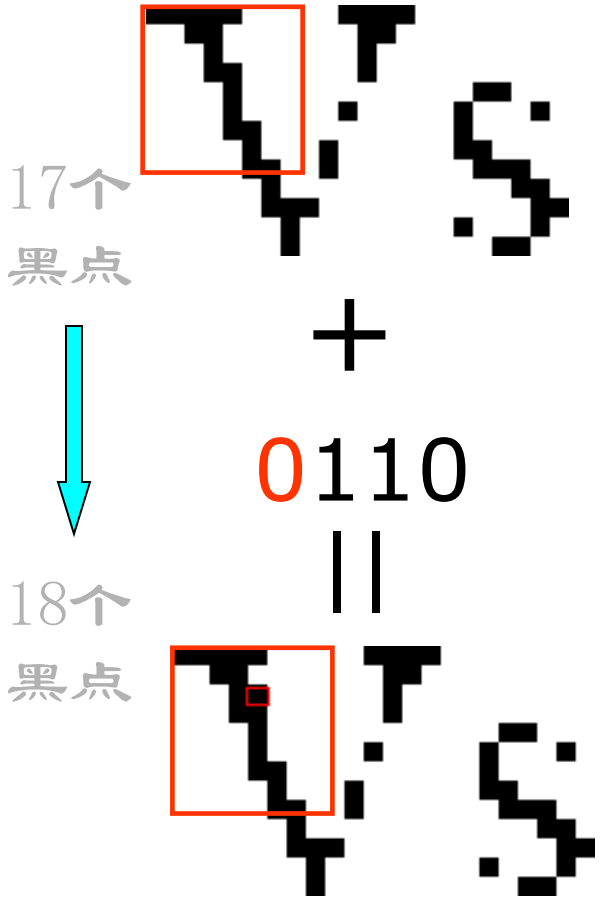
例：游程修改隐藏法

游程的奇偶代表秘密信息1或0



# 2.4 二值图像中的信息隐藏

- 方法三
  - 基于黑色像素个数奇偶性
- 例
  - 8*8小块内黑色像素总个数的奇偶代表秘密信息0或1



# 课程内容

1. 格式信息隐藏算法
2. 时域算法
3. 变换域算法
4. 扩频算法
5. 统计算法

# 3 变换域算法

- 常用的变换域方法
  - 离散余弦变换（DCT）
  - 离散小波变换（DWT）
  - 离散傅立叶变换（DFT）
  - …

# 3 变换域算法

- 扩频通信基本原理
  - 香农得出著名的信道容量计算公式:
  - $C = W log_2\left(1 + \frac{\delta_X^2}{\delta_n^2}\right)$
    - $C$是信道容量
    - $W$是信道带宽
    - $\delta_n^2$是噪声功率
    - $\delta_X^2$是信号功率
  - 香农公式表明信道无误差传输信息的能力与存在于信道中的信噪比以及信道带宽之间的关系

# 3 变换域算法

- 扩频通信基本原理
  - 变换公式为：
  - $\frac{C}{W} = 1.44 log_e \left(1 + \frac{\delta_X^2}{\delta_n^2}\right)$
  - 当$\frac{\delta_X^2}{\delta_n^2} \ll 1$时，用幂级数展开上式，略去高次项
  - $\frac{C}{W} = 1.44 \frac{\delta_X^2}{\delta_n^2}$
  - 得出结论：给定信道容量，可用不同的带宽和信噪比来传输信息。当信噪比较小时（干扰很强），可用宽带系统传输信息。

# 3 变换域算法

- 扩频通信基本原理
  - 扩频通信即扩展频谱通信，信号在大于所需的带宽内进行传输。
- 特点：
  - 占据频带很宽，每个频段上的能量很低
  - 即使几个频段的信号丢失，仍可以恢复信号
- 优势：拦截概率小，抗干扰能力强

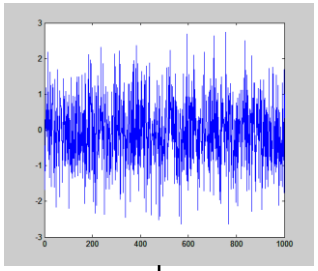
# 3 变换域算法

- 扩频通信基本原理
  - 将扩频通信的概念应用到隐蔽通信系统中。
  - 隐蔽通信系统就是试图将秘密信息扩展在整个载体中，以达到不可察觉的目的，并且删除一小部分载体，也很难删除整个信息。

# 3 变换域算法

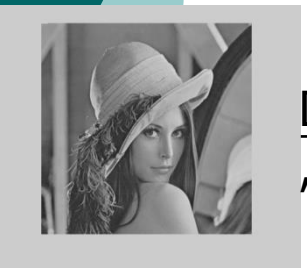
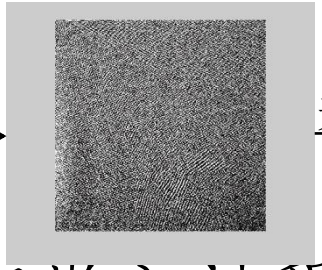
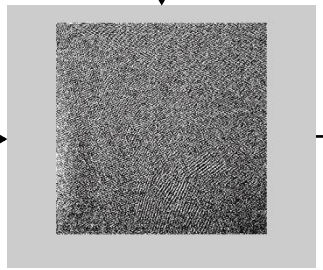
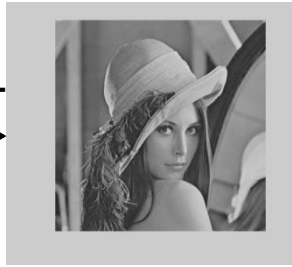
- 基于扩频思想的数字水印算法
  - Cox等人研究发现，在对感官质量影响最关键（或对信号处理和攻击具有很好鲁棒性）的区域嵌入水印，将使水印系统具有较强鲁棒性。
  - 矛盾的是，嵌入带来的、对这些区域的修改，会导致载体感官质量下降。
  - 为了解决上述矛盾，Cox将载体(变换域）视为通信信道，将水印视为信道上传输的信号，将传输过程中水印系统必须抵抗的处理（攻击）视为噪声。类比扩频通信技术，将水印以扩频的方式嵌入到载体最重要的频谱成分中。

# 3 变换域算法



嵌入水印

- 基于扩频思想的数字水印算法


DCT(DFT,DWT)

关键系数

IDCT(IDFT,IDWT)


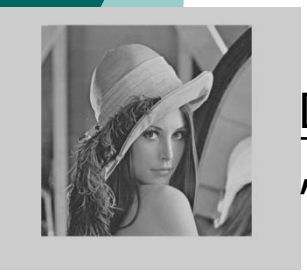
- 水印嵌入过程
  - 产生服从正态分布的长度为N的随机序列作为水印W
  - 选择载体图像N个最大的DCT系数(T)来嵌入水印，如下式:
  - $T_w[i] = T_o[i](1 + \alpha W[i]), i = 1 \cdots N$
  - 对携带水印的系数进行DCT逆变换得到水印载体

# 3 变换域算法

- 基于扩频思想的数字水印算法
  - 思考
    - 水印是嵌入在能量最大的前N个DCT系数，水印嵌入过程可能会导致系数大小发生变化，那么接收端如何得知，应该由哪些DCT系数提取水印呢？
    - 待检测载体是否包含水印，是将提取结果与水印相关，根据相关值进行判决。那么判决阈值如何设定?

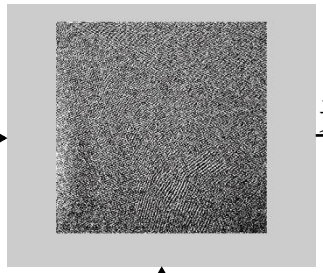
# 3 变换域算法

- 基于扩频思想的数字水印算法

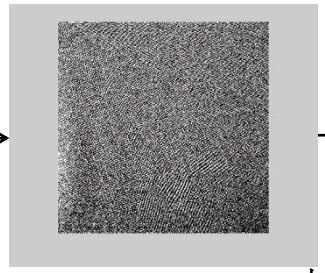


水印载体

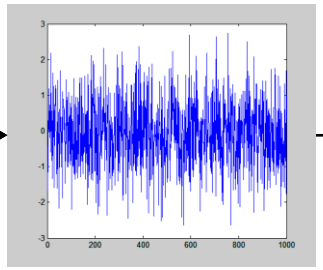
DCT(DFT,DWT)



关键系数



恢复水印



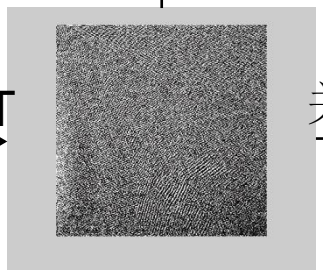
判决结果

根据提取所得水印与原始水印相关性判决

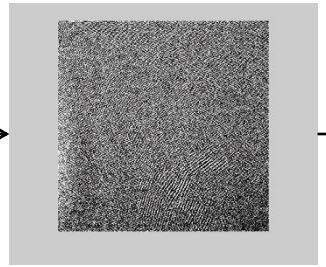
关键系数位置

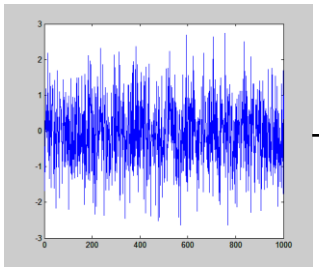
原始载体

DCT(DFT,DWT)



关键系数





原始水印

# 3 变换域算法

- 基于扩频思想的数字水印算法
  - 水印提取过程
    - 对接收到载体（待判决，可能含有水印，也可能没有）做DCT变换，根据原始载体确定水印位置
    - 根据嵌入位置系数提取水印，如下式：
    - $W_n[i] = \left(\frac{T_{wn}[i]}{T_O[i]} - 1\right)/\alpha, i = 1 \cdots N$
    - 考察嵌入水印与提取水印的相关系数，若大于阈值，则认为带检测载体包含水印。
    - $\tau = \sum_i W_n[i] \cdot W[i] / \sqrt{\sum_i W_n[i] \cdot W_n[i]}, i = 1 \cdots N$

# 3 变换域算法

- 基于扩频思想的数字水印的一般模型
  - 水印不仅可以嵌入到DCT系数上，也可嵌入到离散小波、离散傅立叶等等变换系数之上。
  - 嵌入是通过修改系数实现的，系数修改方式可使用下列任一表达式:
    - $T_w[i] = T_o[i](1 + \alpha W[i]), i = 1 \cdots N$
    - $T_w[i] = T_o[i] + \alpha W[i], i = 1 \cdots N$
    - $T_w[i] = T_o[i]\left(e^{\alpha W[i]}\right), i = 1 \cdots N$
  - 水印非盲提取。

# 3 变换域算法

| 水印位置 | 嵌入算法 | 提取算法 |
| --- | --- | --- |
| 所有中频系数 | $x'(i,j)=x(i,j)+\alpha m_i$ | $m'_i=\dfrac{x''(i,j)-x(i,j)}{\alpha}$ |
| 固定位置的中频系数<br>随机挑选中频系数 | $x'(i,j)=x(i,j)(1+\alpha m_i)$ | $m'_i=\dfrac{\frac{x''(i,j)}{x(i,j)}-1}{\alpha}$ |
| 选择最大的N个中频系数 | $x'(i,j)=x(i,j)e^{\alpha m_i}$ | $m'_i=\dfrac{\ln\frac{x''(i,j)}{x(i,j)}}{\alpha}$ |

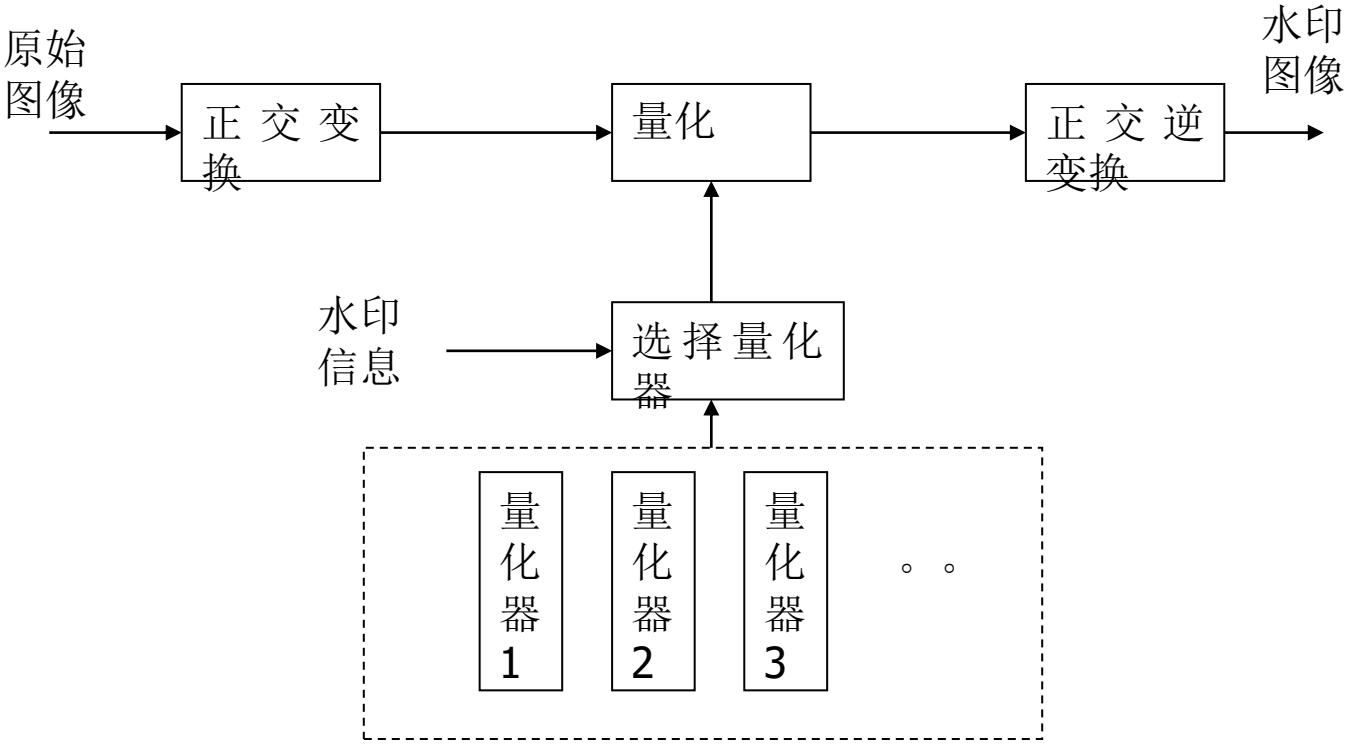
# 3 变换域算法

- 方法一
  - 每个系数上嵌入的强度相同
  - 嵌入强度一定时，
  - 若系数幅值较大，嵌入信息易受破坏
  - 若系数幅值较小，嵌入对系数修改大
- 方法二、三
  - 根据系数的大小，成比例地嵌入
- 这三个算法的提取需要原始图像

# 3.3 量化方法

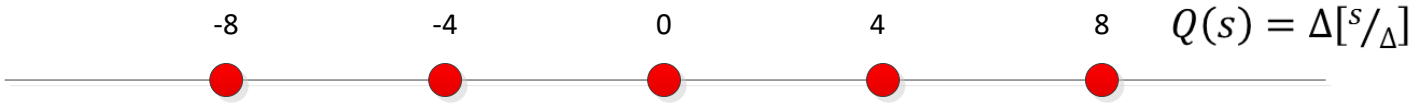
- QIM量化索引调制（Quantization Index Modulation）
- 根据要嵌入的信息，使用不同的量化器对图像进行量化，得到的图像就是含有嵌入信息的图像
- 嵌入信息并没有直接“加”到图像中
- 提取时不需原始图像

# 3.3 量化方法



# 3.3 量化方法(标量量化索引调制)

- 量化函数定义为：
  - $Q(s) = \Delta[s/\Delta]$，([.],round,四舍五入)
  - 例：Δ取为4时，实数轴量化为：



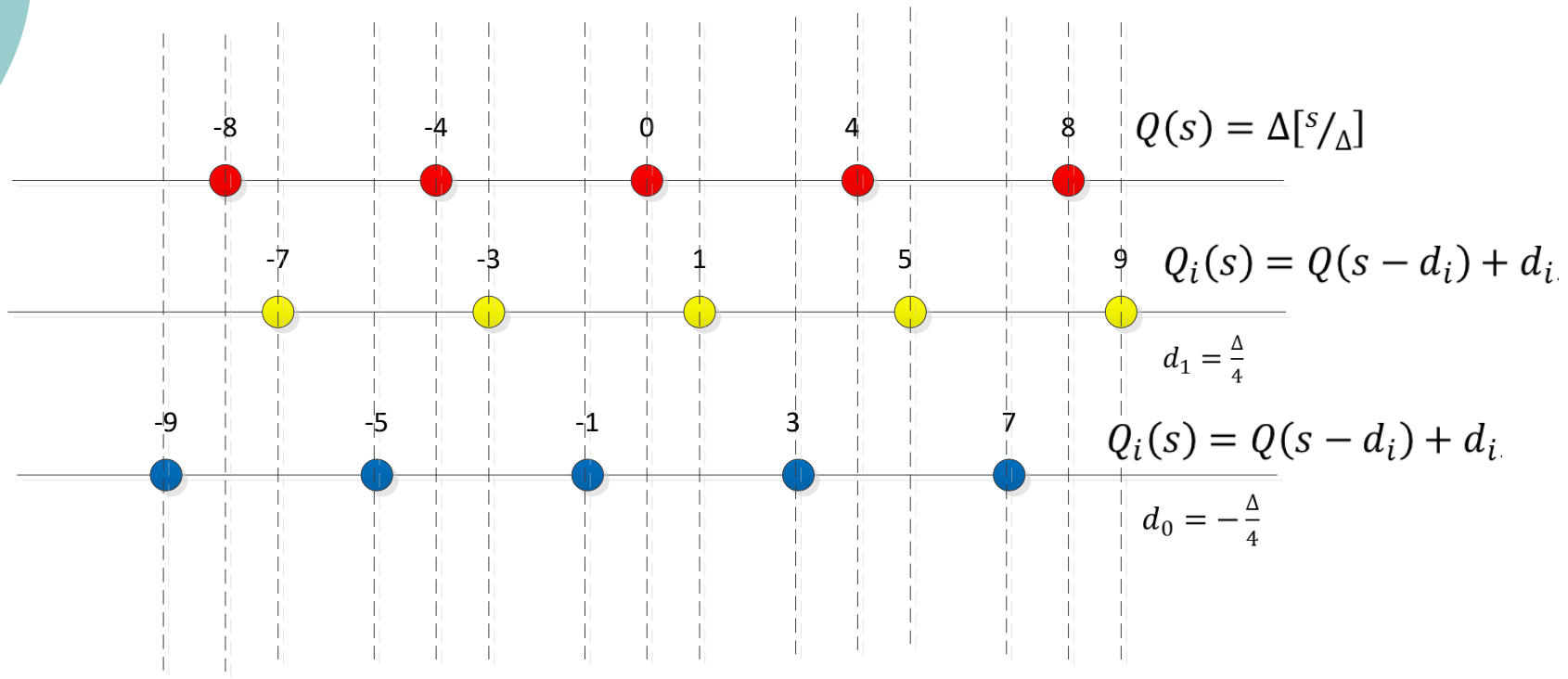

  - 量化点与原始样点之间最大误差是：$\frac{\Delta}{2}$

# 3.3 量化方法(标量量化索引调制)

- 抖动量化器：
  - 为了隐藏二进制数据，至少需要两个量化器。
  - 设计新量化器最简单的方法是平移现有量化器，这种方法称为抖动。
  - 例如：
  - $Q(s) = \Delta[s/\Delta]$，向左和向右各平移$\Delta/4$，可得两个新量化器：
    - $Q(s) = \Delta[s/\Delta]$，$d_0 = -\frac{\Delta}{4}, d_1 = \frac{\Delta}{4}$
    - $Q_i(s) = Q(s - d_i) + d_i, i = 0,1$

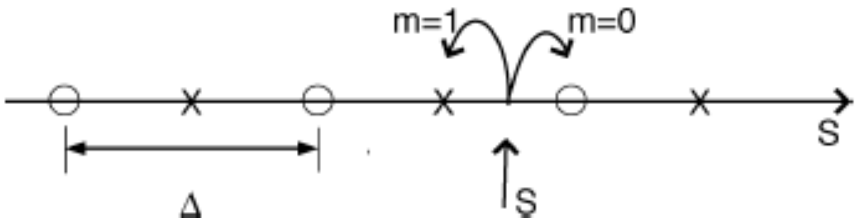
# 3.3 量化方法(标量量化索引调制)

- 当Δ取为4时，抖动量化器为：



# 3.3 量化方法(标量量化索引调制)

- QIM水印算法可以描述为：
  - $Q(x) = \Delta[{}^{x}/_{\Delta}]$，$d_0 = -\frac{\Delta}{4}, d_1 = \frac{\Delta}{4}$
  - $y = Q_{w_i}(x) = Q\left(x - d_{w_i}\right) + d_{w_i}, w_i = 0,1$
  - 其中，$w_i$为待嵌入的信息，$x$为载体对象系数，$y$为隐写对象系数。

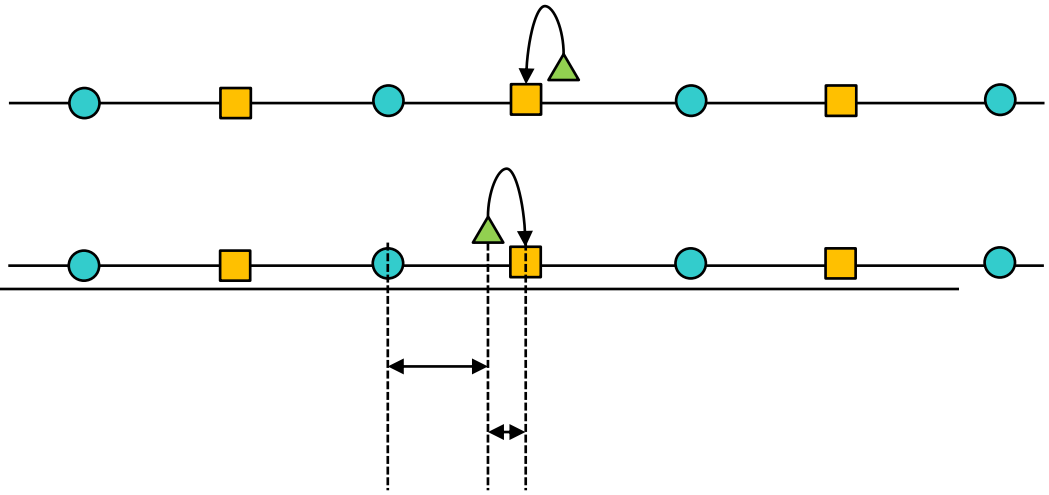



  - 可见，嵌入水印后的系数最大失真为：$\frac{\Delta}{2}$

# 3.3 量化方法

- 量化索引调制
  - 提取水印，即确定相关对象是采用哪一个量化器量化，量化器的索引值对应水印。
  - 然而，嵌入水印时的量化系数，在提取水印时，已经发生了变化。这是因为：
    - 嵌入水印后，量化值经历正交变换逆变换；
    - 水印载体在传输过程中，经受各类信号处理；
    - 提取水印时，水印载体经正交变换。
  - 水印提取有两种方案：最小距离解码和最大似然解码。

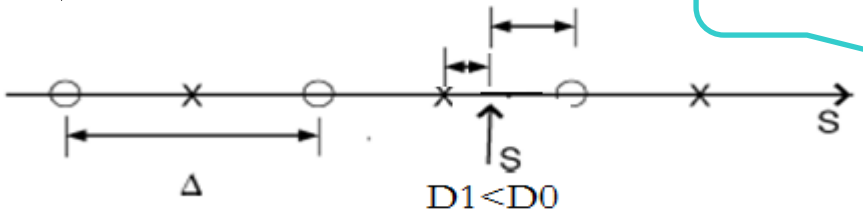
# 3.3 量化方法



- 量化索引调制
  - 最小距离解码：
    - 选择与接收到的数据最接近的量化点作为解码输出，进而确定量化器。
    - 提取按最小距离规则，误差纠正范围是$\left[-\frac{\Delta}{4} \quad \frac{\Delta}{4}\right]$
  - 最大似然解码：
    - 如果载体对象确定，则可知包含水印数据时，接收端收到的数据概率密度，进而可采用最大似然解码。由于概率密度较难准确获取，且最小距离解码实现复杂度低，因此通常使用最小距离解码

# 3.3 量化方法

- 抖动量化索引调制实例分析（提取）
  - 水印提取过程可描述为：
    - $W_n[i] = arg\min_k \|s_{wn} - Q_k(s_{wn})\|$
    - 即：以距离接收数据最近的量化点所对应的量化器索引值为提取的水印。
  - 水印提取效果为：




接收数据与1号量化器的量化值更接近，因此，判决水印为1

  - 思考：
    - 量化步长与算法稳健性有何关系？

# 3.3 量化方法

- 练习
  - 已知量化函数为：
    - $Q(s) = \Delta[s/\Delta]$
    - $Q_{W_0}{}_{[i]}(s) = Q(s - d_{W_0[i]}) + d_{W_0[i]}$,
    - $W_0[i] = 0,1; j = 1 \cdots N$ ， $d_0 = -\frac{\Delta}{4}, d_1 = \frac{\Delta}{4}$
    - 若$\Delta$为2，且已知待量化系数值为2.1，4.0，5.5，4.2，要嵌入的信息为0，1，0，1，则嵌入信息后，系数变为？

# 3.3 量化方法

- 解
  - 要将比特0嵌入系数2.1，则：
    - $Q_0(2.1) = Q(2.1 + 0.5) - 0.5 = 1.5$
  - 要将比特1嵌入系数4.0，则：
    - $Q_1(4.0) = Q(4.0 - 0.5) + 0.5 = 4.5$
  - 要将比特0嵌入系数5.5，则：
    - $Q_0(5.5) = Q(5.5 + 0.5) - 0.5 = 5.5$
  - 要将比特1嵌入系数4.2，则：
    - $Q_1(4.2) = Q(4.2 - 0.5) + 0.5 = 4.5$

# 3.3 量化方法

- 练习
  - 已知量化函数为：
    - $Q(s) = \Delta[s/\Delta]$
    - $Q_{W_0[i]}(s) = Q(s - d_{W_0[i]}) + d_{W_0[i]}$,
    - $W_0[i] = 0,1; j = 1 \cdots N$ ， $d_0 = -\frac{\Delta}{4}, d_1 = \frac{\Delta}{4}$
    - 若$\Delta$为2，且已知系数值2.1，4.0，5.5，4.2嵌入的信息为0，1，0，1后，系数变为1.5，4.5，5.5，4.5，则，若系数没有失真，提取的信息为？若失真分别为：0.1，–0.1，0.2，–0.2，则提取的信息为？

# 3.3 量化方法

- 解：若无失真，
  - 分别假设1.5中嵌入的是0和1，则：
    - $Q_0(1.5) = Q(1.5 + 0.5) - 0.5 = 1.5$
    - $Q_1(1.5) = Q(1.5 - 0.5) + 0.5 = 2.5$
    - 1.5与$Q_0$量化器结果距离相同，所以解码为0。
  - 分别假设4.5中嵌入的是0和1，则：
    - $Q_0(4.5) = Q(4.5 + 0.5) - 0.5 = 5.5$
    - $Q_1(4.5) = Q(4.5 - 0.5) + 0.5 = 4.5$
    - 因为4.5与$Q_1$量化器值距离最近，所以解码为1。

# 3.3 量化方法

- 解：若有失真(5.5+0.2=5.7;4.5-0.2=4.3)，
  - 分别假设5.7中嵌入的是0和1，则：
    - $Q_0(5.7) = Q(5.7 + 0.5) - 0.5 = 5.5$
    - $Q_1(5.7) = Q(5.7 - 0.5) + 0.5 = 6.5$
    - 5.7与$Q_0$量化器结果距离相同，所以解码为0。
  - 分别假设4.3中嵌入的是0和1，则：
    - $Q_0(4.3) = Q(4.3 + 0.5) - 0.5 = 3.5$
    - $Q_1(4.3) = Q(4.3 - 0.5) + 0.5 = 4.5$
    - 因为4.3与$Q_1$量化器值距离最近，所以解码为1。

# 3.3 量化方法

- *Distortion-Compensated Scalar QIM:*
- $Q(s) = \Delta[s/\Delta]$
- $Q_i(s)\begin{cases} = Q_0(\alpha s) + (1-\alpha)s, m = 0 \\ = Q_1(\alpha s) + (1-\alpha)s, m = 1 \end{cases}$
- $\alpha \in [0,1]$

# 3.3 量化方法(矢量量化索引调制)

- 扩展变换抖动调制（STDM）
  - Spread Transform Dither Modulation, STDM
  - 对单一载体数据量化采用的是标量量化的思路。STDM算法，每次对若干载体数据构成的矢量进行量化。
  - 具体方式为：把待量化的矢量投影到某一方向，再对投影数据进行抖动索引量化调制。
  - STDM使得量化失真不再集中在单一的载体数据之上，而是扩展若干载体数据，从而有利于限制峰值失真。

# 3.3 量化方法(矢量量化索引调制)

- 扩展变换抖动调制（STDM）
  - 水印嵌入过程:
  - 选取m个载体数据构成待修改向量$\overline{D_0}$
  - 选取m维向量$\overline{V}$作为投影向量（归一化为单位向量）
  - 计算投影值（$\overline{D_0}$和$\overline{V}$内积）： $s_0 = \overline{V} \cdot \overline{D_0}$
  - 对投影值进行抖动索引量化:
  - $s_w = Q_{W_0[i]}(s_0) = Q(s_0 - d_{W_0[i]}) + d_{W_0[i]}$,
    - $Q(s) = \Delta[s/\Delta]$
    - $W_0[i] = 0,1; j = 1 \cdots N$， $d_0 = -\frac{\Delta}{4}, d_1 = \frac{\Delta}{4}$

# 3.3 量化方法(矢量量化索引调制)

- 扩展变换抖动调制（STDM）
  - 根据量化值重建载体数据向量
  - $\overline{D_w} = \left(\overline{D_o} - s_0\overline{V}\right) + s_w\overline{V}$
  - 对于STDM方法，水印提取过程为：
  - 计算接收数据投影值： $s_{wn} = \overline{V} \cdot \overline{D_{wn}}$
  - 使用最小距离解码：
  - $w_n[i] = arg\min_k \|s_{wn} - Q_k(s_{wn})\|$

# 3.3 量化方法(矢量量化索引调制)

- 练习
  - 请根据STDM算法原理回答问题。
  - 假设投影方向单位矢量为：(1/2, 1/2, 1/2, 1/2)，量化步长为2，那么:
  - 若在两组系数(2,4,3,8)，(4,4,5,10)上嵌入信息”0 1”,那么，这两组系数变为?
  - 若已知使用STDM算法嵌入信息的两组系数为(2,4,3,8)，(4,4,5,10)，那么可提取的信息为?

# 3.3 量化方法(矢量量化索引调制)

- 扩展变换抖动调制（STDM）
  - 由嵌入和提取过程可知，投影向量的选择是关键。满足投影向量是单位向量的前提下，投影向量的常见选择包括:
    - 随机投影向量（随机序列构成的单位向量）
    - 均匀投影向量（各分量值相等的单位向量）
    - 平行投影向量（$\overline{V} = D_0/|D_0|$）
    - 基于频率的投影向量
    - 基于HVS的投影向量

# 3.3 量化方法(矢量量化索引调制)

- 扩展变换抖动调制（STDM）
  - 基于频率的投影向量
    - 在变换域中，修改不同频率系数对稳健性和透明性的影响不同。据此构造投影向量，使得抖动调制只影响我们希望调制的系数，从而获得预期的性能。
    - 例，以$8*8$的DCT变换为例，如果我们只希望影响对角线上系数，则可构建64维投影向量如下：
      - $\overline{X} = (0,0 \cdots 0,1,1 \cdots 1,0,0 \cdots 0)$
      - $\overline{V} = \overline{X}/|\overline{X}|$
      - 投影时，载体DCT变换系数按Zig-Zag扫描

# 3.3 量化方法(矢量量化索引调制)

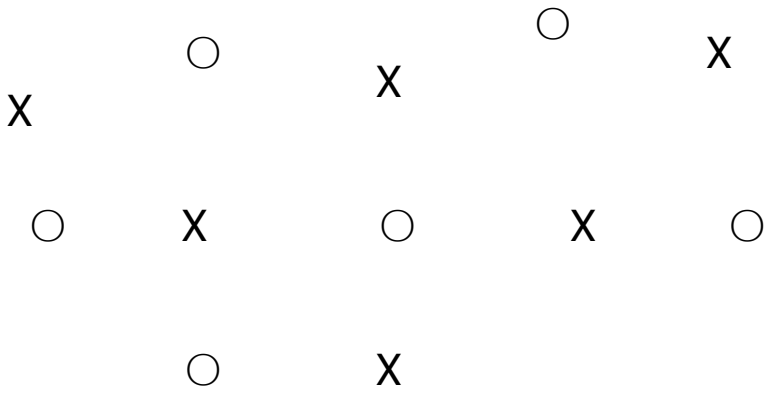
- 扩展变换抖动调制（STDM）
  - 基于视觉模型的投影向量
    - 类似于HAS会根据输入频率和声强的不同产生不同的响应，HAS的响应也会因空间频率、亮度和颜色的不同而产生差异。视觉模型实质上评估噪声对感知影响。
    - 例如waston模型给出了各个DCT系数在一个JND范围内可进行的变化大小。
    - 可以据此构造基于视觉模型的投影向量

$$X[k]=\begin{cases}1 & \text{当}\varphi[k]>T\ (\varphi[k]\text{为一个JND范围内可进行的变化大小})\\ 0 & \text{其他}\end{cases}$$

- $\overline{V}=\overline{X}/|\overline{X}|$

# 3.3 量化方法(矢量量化索引调制)

- 矢量量化示意图




- X：量化器1
- O：量化器2

# 3.3 量化方法(矢量量化索引调制)

- m=1：信号被“X”量化器量化，即：
  信号值用离“X”最近的量化器值代替
- m=0：信号被“O”量化器量化，即：
  信号值用离“O”最近的量化器值代替

# 3.3 量化方法(矢量量化索引调制)

- 提取
  - 设定一个门限
  - 系数值与量化器X的距离小于门限：m＝1
  - 系数值与量化器O的距离小于门限：m＝0

# 小结

- 变换域隐藏的总体思想，就是将秘密信息隐藏在载体的最重要部位
- DCT变换、小波变换、傅氏变换等，都是能量守恒变换，在变换域中将能量集中，隐藏时将秘密信息与载体的视觉重要部分紧密联系在一起

# 课程内容

- 格式信息隐藏算法
- 时域算法
- 变换域算法
- 扩频算法
- 统计算法

# 课程内容

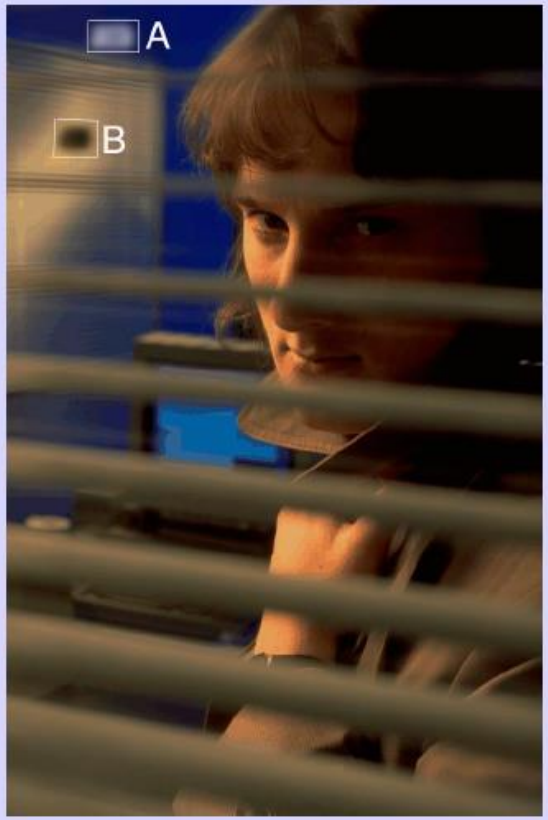
- 格式信息隐藏算法
- 时域算法
- 变换域算法
- 扩频算法
- 统计算法

# 统计隐藏技术

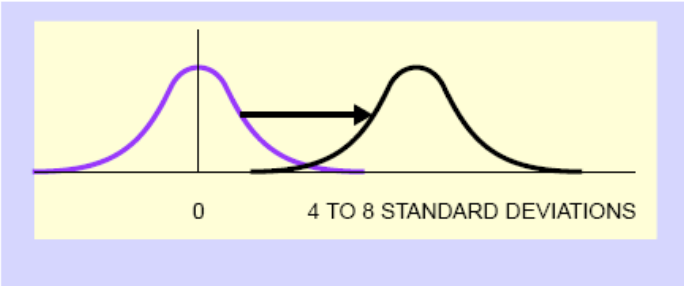
- 原理
  - 对载体的某些统计特性进行明显的修改，表示嵌入信息“1”，
  - 若统计特性不变，则表示嵌入信息“0”。
- 接收者在不知道原始载体的情况下，根据统计特性的改变，提取信息。

# patchwork

- 1995, Bender等人。
- 随机选取两个区域A、B，增加A的亮度，降低B的亮度。
- 根据A、B区域亮度差值判定水印有无



# Patchwork原理



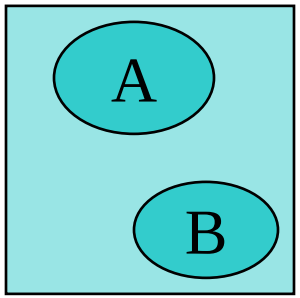

- 原始图像，n对像素点的亮度差值的期望为零
- 经patchwork算法处理后，亮度差值的期望不再为零

# Patchwork算法

- 水印嵌入位置的选择（时空域）
  - 选择视觉不敏感区域（如纹理区域等）
  - Patchwork算法：根据密钥随机选择n个像素对，更改它们的亮度值

$$\tilde{a}_i = a_i + 1$$

$$\tilde{b}_i = b_i - 1$$




$(a_1,b_1)$
$(a_2,b_2)$
$(a_3,b_3)$

$(a_n,b_n)$

$A=\{a_{1,\ldots,}\ a_n\}$

$B=\{b_{1,\ldots,}\ a_n\}$

  - 提取时，计算 $s = \sum_{i=1}^{n} (\tilde{a}_i - \tilde{b}_i)$

# 载体生成技术

- 前面介绍的所有信息隐藏方法，都是修改载体，将秘密信息嵌入。
- 载体生成技术：利用秘密信息，生成一个掩蔽载体，而生成的掩蔽载体是有意义的数字文件，不会引起怀疑，这样也达到了秘密信息传递的目的 。

# 载体生成技术的应用

- 由于信息传输量的爆炸性增长，人类不可能观察世界上所有的通信，这样的任务只能由自动监控系统完成。
- 信息监控系统
  - 通过设置关键字来检查信息，也可以通过统计特性来分析消息的特性，通过监控系统的筛查后，再对可疑信息进行人工检查

# 载体生成技术的应用

- 信息监控系统
  - 例如，对于明文传输的消息，用关键字的方法就可以筛选出某类消息；而对于加密的信息，可以从统计特性上区分加密文本和未加密文本。

# 模拟函数的方法

- 根据秘密信息，
- 使用模拟函数产生的文本，
- 文本统计特征与普通文本近似

# 模拟函数的方法


This is a test
encode
decode
reset form
empty form
encodes to. . .
The brush subtly kicks to the bright square. I kill strange
frogs near the lazy closed road. Sometimes, frogs restrain
behind tall roofs, unless they're squishy. Never lean fully
while you're buying through a quiet unit. We wanly.
encode
decode
reset form
empty form

# 模拟函数的方法

- 采用模拟函数产生的文本，由于它是根据统计特性创建的，完全忽略了语义成分，因此，它仅能**欺骗**利用统计特性分析文本的**自动检测机器**，如果让人来看，这个文本是完全没有正常含义的。

# 载体生成技术的应用

- 为了使产生的掩蔽载体既能欺骗机器，又能欺骗人，就是说既有正常的英文统计特性，又有正常的语法含义，看起来是一篇正常的文章，提出了一种英语文本的自动生成技术:自由上下文语法（CFG） 。

# CFG

- G=〈V，Σ，Π，S〉
- V是一个变量集
- Σ是终止符号集
- S∈V是起始符号
- $\Pi \subseteq V \times (V \cup \sum)^*$是叉积。叉积可以看成是一个替代规则，它们把一个变量转化成包含结束符号或变量符号的字符串

# 自由上下文语法（例）

∏={S→Alice B, S→Bob B, S→Eve B, S→I A,
A→am working, A→am lazy, A→am tired,
B→is C, B→can cook,
C→reading, C→sleeping, C→working}

# 自由上下文语法(例)

$\prod$={

S$\rightarrow_{0.5}$Alice B, S$\rightarrow_{0.3}$Bob B, S$\rightarrow_{0.1}$Eve B, S$\rightarrow_{0.1}$I A,

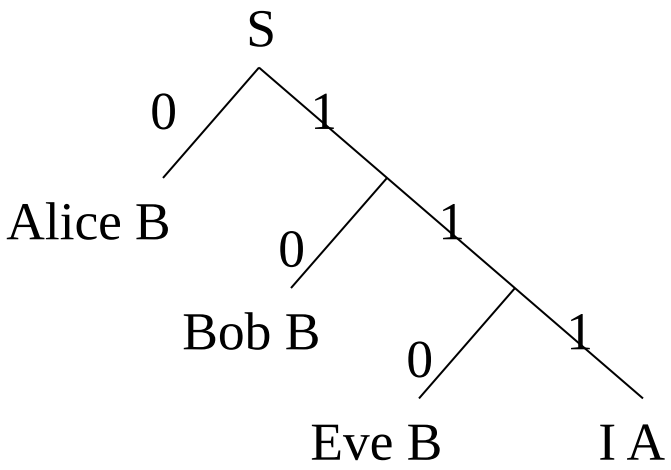
A$\rightarrow_{0.3}$am working, A$\rightarrow_{0.4}$am lazy, A$\rightarrow_{0.3}$am tired,

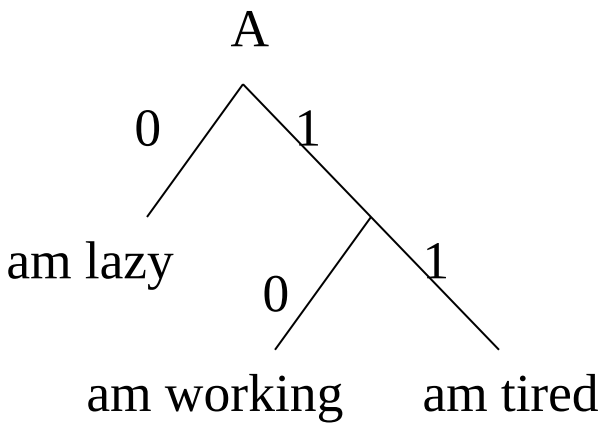
B$\rightarrow_{0.5}$is C, B$\rightarrow_{0.5}$can cook,

C$\rightarrow_{0.5}$reading, C$\rightarrow_{0.1}$sleeping, C$\rightarrow_{0.4}$working

}

# 哈夫曼编码




“Eve B”编码为110
“I A”编码为111

“A am tired”编码为11

“I am tired” 11111

# 基础练习

- 1. 有关基于格式的信息隐藏技术，下列描述不正确的是:
  - A.隐藏内容可以存放到图像文件的任何位置。
  - B.隐藏效果好，图像感观质量不会发生任何变化。
  - C.文件的拷贝不会对隐藏的信息造成破坏，但保存文档时可能会造成隐藏数据的丢失。
  - D.隐藏的信息较容易被发现。

# 基础练习

- 2. 如果对调色板图像像素采用LSB方法进行处理以隐藏数据，下列描述不正确的是

  A. 索引值相邻的颜色对，其色彩或灰度可能相差很大，因此替换后图像感观质量可能会有明显下降。

  B. 图像处理软件可能会根据颜色出现频率等重排颜色索引，因此隐藏的信息可能会丢失。

# 基础练习

- 2. 如果对调色板图像像素采用LSB方法进行处理以隐藏数据，下列描述不正确的是

  C. 方法的优点是可隐藏的数据量大，不受图像文件大小限制。

  D. 为防止索引值相邻的颜色对色差过大，可以根据其色度或灰度预先进行排序，改变索引顺序，再对像素进行LSB替换。

# 基础练习

- 3. 已知某图像轮廓的游程编码为：<a0, 3><a1, 4><a2, 4><a3, 7>。现需修改游程长度以隐藏秘密信息，约定隐藏0时游程长度为偶数（约定长度在2i和2i+1之间翻转，例如2-3, 4-5,...），则隐藏秘密信息1100后，游程编码变为：
  - A.<a0,3><a1,5><a2+1,2><a3-1,8>
  - B.<a0,3><a1,5><a2,2><a3,8>

# 基础练习

- 3. 已知某图像轮廓的游程编码为：<a0, 3><a1, 4><a2, 4><a3, 7>。现需修改游程长度以隐藏秘密信息，约定隐藏0时游程长度为偶数（约定长度在2i和2i+1之间翻转，例如2-3, 4-5,...），则隐藏秘密信息1100后，游程编码变为：

  C.<a0,5><a1+2,5><a2+2,4><a3+2,8>

  D.<a0,5><a1+2,3><a2+1,4><a3+1,8>

# 基础练习

- 4. 现接收到一使用DCT系数相对关系（隐藏1时，令B(u1，v1)>B(u3, v3) +D,且，B(u2, v2)>B(u3, v3)+D）隐藏秘密信息的图像，已知D=0.5，对该图像作DCT变换后，得到约定位置((u1，v1)(u2, v2)(u3, v3))的系数值为：(1.6, 2.1, 1.0)，（0.7, 1.2, 1.8）,(0.9, 1.8, 1.2)，则可从中提取的秘密信息是:

# 基础练习

- 4. 已知D=0.5，对该图像作DCT变换后，得到约定位置的系数值为：(1.6, 2.1, 1.0)，（0.7, 1.2, 1.8）,(0.9, 1.8, 1.2)，则可从中提取的秘密信息是:
- A. 0,1,1
- B. 1,0,0
- C. 1,0,无效
- D. 0,1,无效

# 基础练习

- 5. 已知某调色板图像使用基于索引的信息隐藏算法嵌入了秘密信息，该调色板图像共有3种颜色，颜色的标准排序是c0,c1,c2，而接收图像的颜色排序为c1,c0,c2。则可从中提取的秘密比特是:
  - A 11 B 10 C 01 D无效