**一、盲水印定义与特点**

**盲水印（Blind Watermarking）是一种无需原始图像**即可从被篡改图像中**提取水印信息**的数字水印技术，具备以下特点：

* 无需原图参与提取；
* 对攻击（裁剪、压缩、旋转等）具有鲁棒性；
* 适合版权追踪、泄露检测、安全认证场景。

**二、嵌入与提取的基本框架**

设：

* I：原始图像（二维像素矩阵）；
* w：水印信息（通常为二进制序列）；
* I\_w：嵌入水印后的图像；
* k：密钥或伪随机种子（控制水印位置与强度）；

则水印嵌入和提取可形式化为：

**嵌入函数：**



**提取函数：**



**三、核心算法原理**

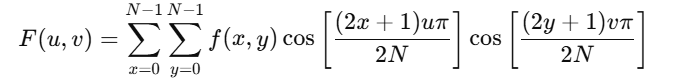
在盲水印中，一般使用**图像频域系数**或**统计特征**嵌入水印信息，常见方式如下：

**1. DCT 域盲水印原理（常见方式）**

* 将图像分块并进行**离散余弦变换（DCT）**；
* 在中频系数中嵌入水印位，控制强度；
* 逆变换恢复图像。

**步骤：**

1. **DCT变换：**



1. **水印嵌入方式（QIM量化索引调制）：**

对某个 DCT 系数 ccc，根据水印位 b∈{0,1}b \in \{0, 1\}b∈{0,1}，进行量化嵌入：

黑色的钟表

AI 生成的内容可能不正确。

其中 Q 为量化步长，控制鲁棒性与不可见性权衡。

1. **提取水印：**

文本

AI 生成的内容可能不正确。

**2. DWT 域盲水印原理（小波域）**

* 对图像进行小波分解（如 Haar、Daubechies）；
* 在 LH、HL、HH 子带中嵌入水印；
* 利用子带的冗余性增强抗攻击能力。

**水印嵌入（简化）：**



其中：

* C：小波系数矩阵；
* w：水印比特序列（调制为 +1 或 -1）；
* α\alphaα：嵌入强度因子。

**提取：**

文本

AI 生成的内容可能不正确。​​

在盲水印中，C\_估计​ 可由图像的局部均值或邻域推断。

**四、密钥机制与安全性**

盲水印通常引入密钥 kkk，控制水印的：

* 嵌入位置（伪随机位选取）；
* 嵌入顺序（混淆顺序）；
* 编码方式（如扩频序列）；

**常用方法：**

* 伪随机序列生成器 PRNG（种子为密钥）；
* XOR 加密水印数据；
* 哈希函数生成位置表。

**五、鲁棒性保障机制**

为了增强抵抗图像处理攻击（如裁剪、旋转、缩放、滤波、压缩等）的能力，盲水印通常结合以下策略：

* **冗余嵌入**：重复嵌入多个区域；
* **容错编码**：使用 Hamming / BCH / RS 编码增强抗错能力；
* **频域嵌入**：在对抗性较强的频域系数嵌入；
* **区域匹配**：攻击后使用模板匹配还原位置；
* **盲估计参数恢复**：如裁剪位置、缩放因子的估计；

代码结构说明：

Blind\_watermark：开源项目源代码

Examples/run.py: 鲁棒性测试代码

开源项目中没有给出对比度和平移攻击的测试代码，补充如下：

文本

AI 生成的内容可能不正确。

文本

AI 生成的内容可能不正确。

平移攻击也就是对加了水印的代码进行水平位置的偏移，提取时会出现错位解码或无法解码

对比度调整通过拉伸或压缩图像像素值的范围，改变图像亮暗差异，扰乱水印所依赖的灰度分布统计特性。若提取过程对图像像素敏感，可能导致信息丢失或误码

运行run.py

文本

AI 生成的内容可能不正确。

可以看到这种盲水印方法对平移和对比度攻击是不具备还原水印的功能的，攻击会成功