基于奇异值的数字水印 matlab 实现

1.实验目的

- 检索和分析相关文献，了解基于奇异值的数字水印.
- 用 matlab 实现基于奇异值的数字水印仿真测试.
- 对实验设计的算法进行性能分析.

2.实验环境

- 使用的载体图像为 512x512 的 lena 灰度图像，使用的工具及版本为 MATLAB R2017b.

3.发展现状

数值分析中的奇异值分解(SVD) 是一种将矩阵对角化的数值算法. 在图像处理中应用 SVD 的主要理论背景是: (1) 图像奇异值的稳定性非常好, 即当图像被施加小的扰动时, 图像的奇异值不会有大的变化; (2)奇异值不是反映图像的视觉特性,而是反映图像矩阵元素间的关系,即图像的内蕴特性。

基于 SVD 的数字水印算法由刘瑞桢、谭铁牛 2001 在《基于奇异值分解的数字图像水印方法》中最先提出，该算法首先对图像做 SVD 分解，然后通过将水印信息矩阵加载奇异值矩阵上来嵌入水印，该算法的结果比当时通用的 COX 方法鲁棒性也好很多.但该算法在检测水印时需要用到关于原始载体图像的 3 个矩阵，所需信息量较大.

2002 年孙锐、孙洪、姚天任在《基于奇异值分解的半易损水印算法》也提出了一种基于奇异值分解的半易损水印算法，该算法是根据失真要求确定一个量化因子，然后将块最大奇异值相对于此量化因子取模，最后根据模值和水印信息来修改最大奇异值。因为水印信号是通过随机排序嵌入最大的奇异值中，使该算法具有较好的安全性与稳健性，嵌入水印后的图像保持了较好的品质。而且该算法使用了量化策略嵌入水印，使得提取水印信号无需使用原始图像。实验结果表明该算法可将 JPEG 有损压缩同恶意攻击区分开来，准确的定位被篡改的图像内容，对常用的图像增强算法也具有较强的稳健性和较好的适用性，但此方法对其他图像处理攻击的鲁棒性并非十分理想.

2003 年胡志刚、谢萍、张宪民在《一种基于奇异值分解的数字水印算法》中提出了一种基于 SVD 的水印算法，该算法通过对块奇异值组成的向量求范数，然后对范数进行量化，再根据水印信息修改量化值的奇偶性，从而嵌入水印。从攻击实验结果可知，该算法对常用的信号处理具有较好的鲁棒性，而且具有透明性好、水印检测结果准确、水印的提取不需要原图像即盲检测等特点。但是该算法有一不足点就是嵌入失真比较大，尤其是在需要修改量化值的情况下.

同年张志明、李蓉艳、王磊在《基于混沌变换的 SVD 图像水印算法》也提出了一种基于混沌变换的 SVD 图像水印算法，该算法通过在水印嵌入以前对水印图像进行混沌置换加密增强了算法的安全性，而且水印嵌入过程应用了 SVD 块分类方法结合 HVS 对嵌入强度取值实现水印分量的自适应嵌入，获得了更好的视觉掩蔽性。即根据分块 SVD 得到的奇异值序列以及奇异向量对判断分块所属类别选取水印强度因子，这样水印信号就会加入到载体图像的低频部分也就是分块奇异值的最大值分量里。实验结果也表明这种自适应方法增强了算法的稳健性.

2005 年张晓梅在《基于奇异值分解和扩频技术的数字水印算法研究》提出了一种全新的基于奇异值分解和扩频技术的数字水印算法。该算法在水印嵌入之前先对水印信息进行置乱,并利用置乱位置作为密钥产生扩频码序列来对置乱后的水印信息进行频谱扩展,这样的预处理可以提高水印信号的保密性,增强水印抵抗恶意攻击的能力。然后将扩频后的水印信息加在载体图像的奇异值分解域中。这样虽然水印原始图像是有意义的,但是嵌入载体图像的水印信号却是无意义的随机序列,极大的提高了系统的保密性。

2006 陈帆、和红杰、朱大勇也提出了一种基于图像奇异值的脆弱水印算法，该算法是选取与图像内容密切相关的图像奇异值作为水印，用混沌系统对其加密来增强水印的安全性的。认证时则通过水印差值定位图像被篡改的位置，通过奇异值差值反映被篡改区域的篡改强度。因为根据矩阵奇异值分解的性质，图像的奇异值具有良好的稳定性，因此奇异值的变化范围可以在一定程度上反映篡改区域被篡改的强度.

同年张志明、周学广也提出了一种基于奇异值分解的水印算法，该算法是利用原图像 SVD 分解后得到的正交矩阵为容器进行水印嵌入的，而且在水印嵌入过程中使用序列密码加强了水印的保密性，同时使用汉明纠错码和算法的迭代保证了水印在提取时有较强的鲁棒性.

2009 年刘润涛、孙中喜、倪金霞、周洪玉在《基于奇异值分解的小波域灰度数字水印算法》一文中，针对灰度数字水印的隐藏问题,依据 Arnold 变换、奇异值分解及小波分析理论,提出了一种基于奇异值分解的小波域灰度数字水印算法,该算法对图像先进行二级小波分解,将分解得到的低频子图进行奇异值分解,然后将经 Arnold 变换置乱后的灰度水印嵌入.该算法有效克服了灰度水印数据量大的缺点.

2011 年徐慕蓉、樊锁海为提高图像和水印的鲁棒性,根据奇异值分解的特性,结合 Aronld 变换的分散攻击和人类视觉系统寻找最优嵌入强度的良好性能,提出了一种新的自适应图像数字水印算法。首先将图像分块,然后计算得到每块图像的最大奇异值,再将经 Arnold 变换后的水印结合通过人类视觉系统技术得到的水印强度嵌入最大奇异值,从而得到带水印的图像。

4. 实验参考资料

英文：LIU Rui-zhen and TAN Tie-niu,SVD Based Digital Watermarking Method, in Acta Electronica Sinica,vol. 29,no. 2,pp.168-171,Feb 2001.(cite 200+)

中文：《基于奇异值分解的数字图像水印方法》 ---刘瑞桢，谭铁牛

5.实验原理

- 奇异值分解基本原理分析
  - 一幅灰度图像可以被看成是一个非负矩阵，对于 N×N 的矩阵 A，有 $\lambda_i(i = 1,2,\cdots,N)$ 个标量 满足：

$$|A - \lambda_i I| = 0$$

则称这一组 λ_i为矩阵 A 唯一的特征值.

  - 如果存在这样一个 N×1 的向量V_i,有:

$$AV_i = \lambda_i V_i$$

则称V_i为 A 的与特征值λ_i对应的一个特征向量. A 一共有 N 个特征向量.

  - 矩阵的奇异值分解(Singular Value Decomposition,简称 SVD）是矩阵所固有的特征，设矩阵 $A \in R^{m\times n}$，rank(A)=r，r≤n，那么矩阵 的奇异值分解定义如下：

$$A = UDV^T = [u_1,u_2,\cdots,u_m]\begin{bmatrix}\sigma_1 0\cdots 0\\ \vdots\\ 00\cdots\sigma_r\end{bmatrix}_{m\times n}[v_1,v_2,\cdots,v_n]^T = \sum_{j=1}^{r}\sigma_j u_j v_j$$

其中，$U = [u_1,\cdots u_m] \in R^{m\times m}$和 $V = [v_1,\cdots v_n] \in R^{n\times n}$是正交矩阵，其列向量分别为$u_i$和$v_i$；U，V 分别称为矩阵 A 的左奇异矩阵和右奇异矩阵；D 是对角阵；$\sigma_i(i = 1,\cdots,r)$称作矩阵 A 的奇异值，此处是 AA^T或A^TA 的特征值λ_i的正平方根，满足：

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > \sigma_{r+1} = \cdots = \sigma_m = 0$$

- 水印的嵌入
  - 采用的基本思想是将水印嵌入到图像矩阵的奇异值中，在水印的嵌入过程中，先做 m×n 灰度图像 A 的奇异值分解，得到两个正交矩阵 U、V 及一个对角矩阵 S. 水印 $W \in R^{n\times n}$ 被叠加到矩阵 S 上，对新产生的矩阵 $S+\alpha W$ 进行奇异值分解，得到U_1、S_1和$V_1(S+\alpha W = U_1 S_1 V_1^T)$，其中常数$\alpha > 0$ 调节水印的叠加强度。然后将矩阵 U、S_1和V^T相乘，得到处理后的包含水印的图像$\hat{A}$。
  - 具体步骤：

    即如果矩阵 A 和 W 分别表示原始图像和水印，那么通过如下三个步骤得到水印图像$\hat{A}$:

    1. 将载体图像矩阵 A 进行奇异值分解：

    $$A = USV^T$$

    2. 将水印图像 W 叠加到对角阵 S 得到一个新的矩阵，再对新矩阵进行奇异值分解：

    $$S' = S + \alpha W$$

    $$S' = U_1 S_1 V_1^T$$

    3. 得到含水印的图像$\hat{A}$

    $$\hat{A} = US_1V^T$$

- 水印的提取
  - 在水印的检测过程中，如果给出矩阵U_1、S、V_1和可能损坏的水印图像A^*，那么通过简单的逆过程就就可以提取出可能已经失真的水印 W^*:

    1. 可能损坏的水印图像A^* 进行奇异值分解：

    $$A^* = U^* S_1^* V^{*T}$$

2. 计算中间矩阵：

$$D^* = U_1 S_1^* V_1^T$$

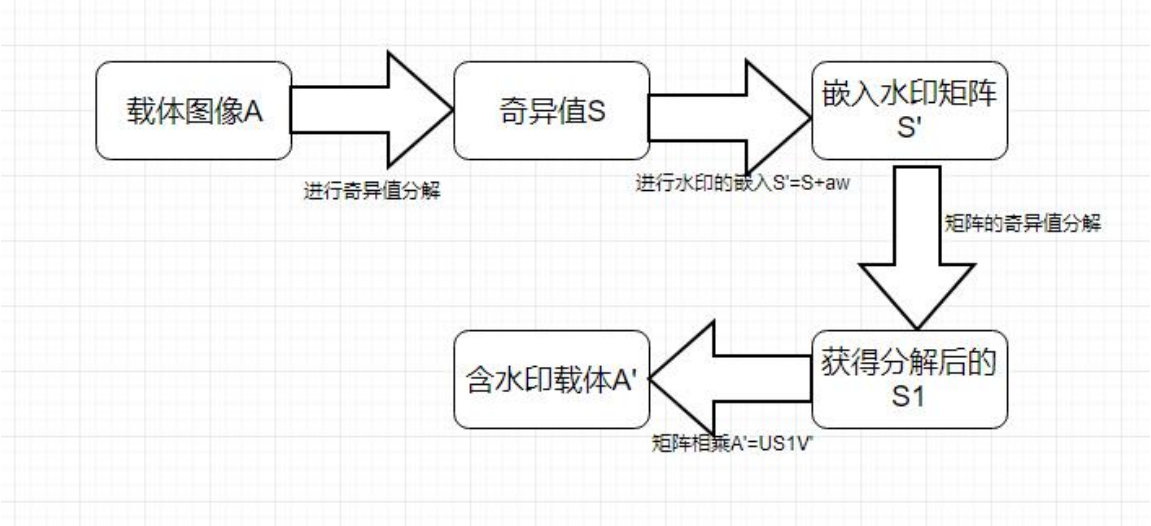
3. 获得水印图像：

$$W^* = \frac{1}{\alpha}(D^* - S)$$

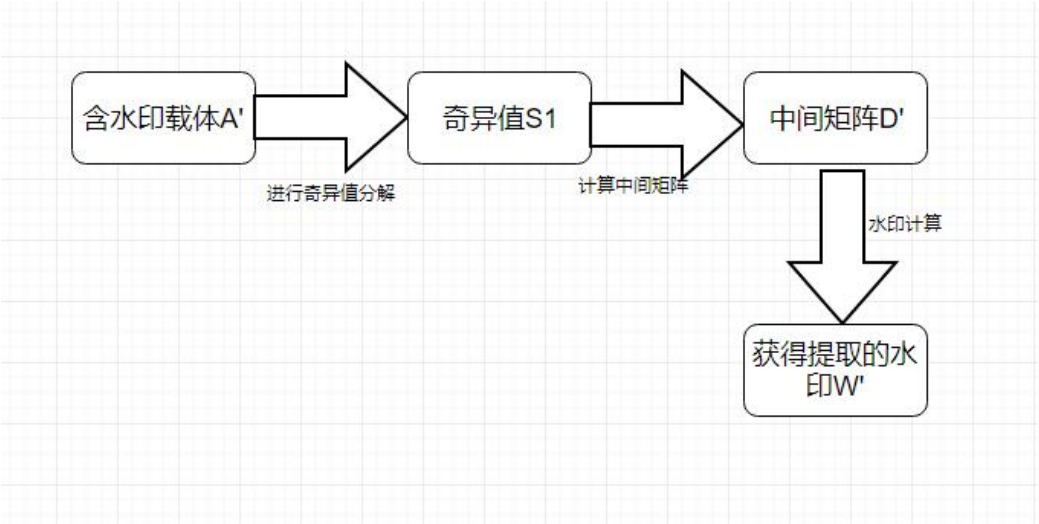
4. 计算提取的水印与原水印的相似系数，作为二者的相似程度，相似系数越接近 1 就越相似。

6.实验过程

- 水印嵌入流程图



- 水印提取流程图



- 水印算法性能测试

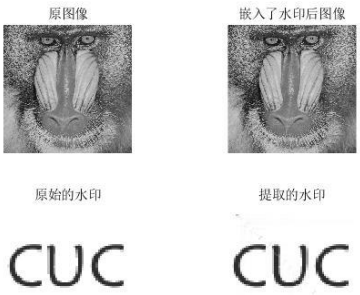
测试方法	参数
不同容量嵌入	嵌入图像大小从 64×64 到 114×114 进行嵌入
叠加高斯噪声	均值为 0，方差不同(0.01、0.02、0.03、…0.50)
叠加椒盐噪声	噪声密度不同(0.01、0.02、0.03…0.50)
均值滤波	均值滤波程度
叠加高斯噪声及滤波	均值为 0，方差为 0.005，添加均值及加权均值滤波
叠加椒盐噪声及滤波	噪声密度为 0.100，添加均值、中值及加权均值滤波
JPEG 压缩	压缩质量因子 1~100 对图像进行批量压缩
篡改图像内容	对图像进行旋转、添加素材、裁剪
直方图均衡化	
调节水印的叠加强度	叠加强度为 0.0001、0.001…1…100

7.实验结果

- 对 lena 图像进行嵌入与提取

相关系数：0.9988

- 对其余 20 幅图像进行嵌入与提取

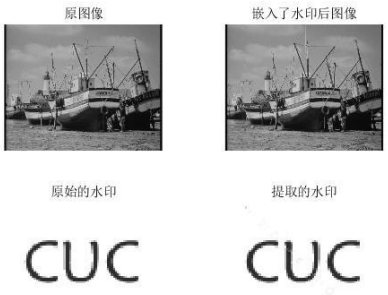


相关系数：0.9963

PSNR：99.2810

相关系数：0.9957

PSNR：104.3577



相关系数：0.9950

PSNR：103.4005

相关系数：0.9969

PSNR：101.2227

原图像 嵌入了水印后图像

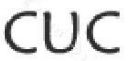

原始的水印 提取的水印


相关系数：0.9959

PSNR：93.2661

原图像 嵌入了水印后图像


原始的水印 提取的水印

相关系数：0.9956

PSNR：101.0870

原图像 嵌入了水印后图像

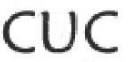

原始的水印 提取的水印


相关系数：0.9969

PSNR：102.0853

原图像 嵌入了水印后图像


原始的水印 提取的水印

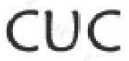



相关系数：0.9935

PSNR：101.0960

原图像 嵌入了水印后图像

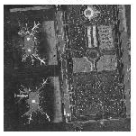

原始的水印 提取的水印

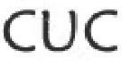
相关系数：0.9962

PSNR：101.6274

原图像 嵌入了水印后图像




原始的水印 提取的水印

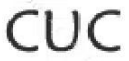



相关系数：0.9956

PSNR：100.2639

原图像 嵌入了水印后图像

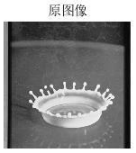
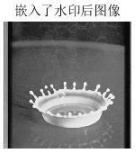

原始的水印 提取的水印

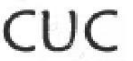
相关系数：0.9948

PSNR：101.6579

原图像 嵌入了水印后图像

原始的水印 提取的水印

相关系数：0.9818

PSNR：101.2201

相关系数：0.9955

PSNR：102.3870

相关系数：0.9954

PSNR：102.1099

相关系数：0.9946

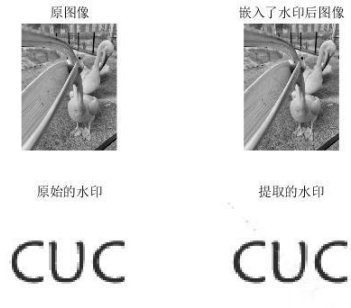
PSNR：101.4719

相关系数：0.9860

PSNR：101.4741

相关系数：0.9898

PSNR：111.1950



相关系数：0.9947

PSNR：111.3785

8. 性能测试

- 以下实验进行了对准确度的设定，即使用 corrcoef 函数（计算相关系数），如下进行了阈值的设定阈值的设定，来源为 matlab 官方文档

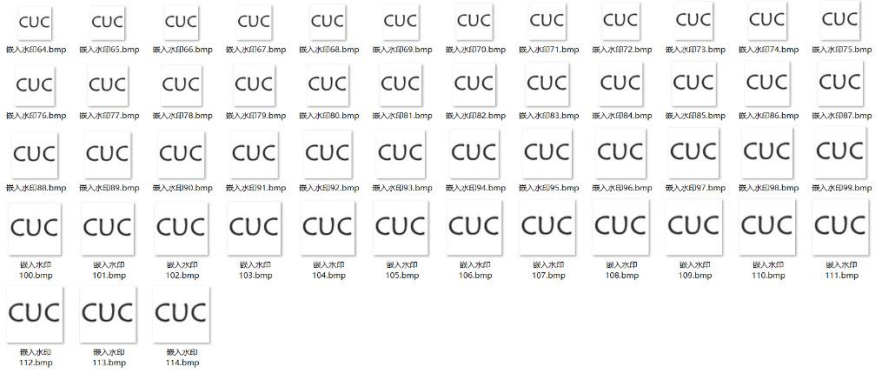
Corrcoef:相关系数：[R,P] = corrcoef(___) 返回相关系数的矩阵和 p 值矩阵，用于测试观测到的现象之间没有关系的假设（原假设）。此语法可与上述语法中的任何参数结合使用。如果 P 的非对角线元素小于显著性水平（默认值为 0.05），

则 R 中的相应相关性被视为显著。如果 R 包含复数元素，则此语法无效。因此后续的阈值设定为 0.05

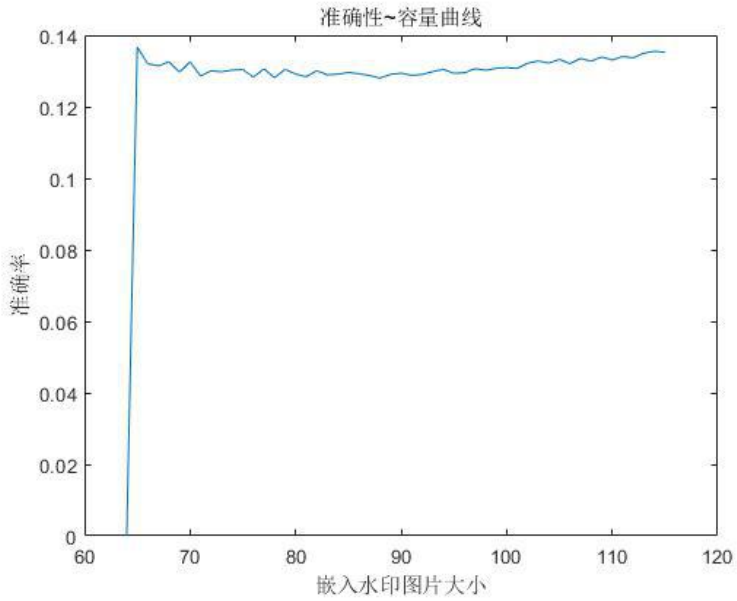
- 测试方法及参数

测试方法	参数
不同容量嵌入	嵌入图像大小从 64×64 到 114×114 进行嵌入
叠加高斯噪声	均值为 0，方差不同(0.01、0.02、0.03、…0.50)
叠加椒盐噪声	噪声密度不同(0.01、0.02、0.03…0.50)
均值滤波	均值滤波程度
叠加高斯噪声及滤波	均值为 0，方差为 0.005，添加均值及加权均值滤波
叠加椒盐噪声及滤波	噪声密度为 0.100，添加均值、中值及加权均值滤波
JPEG 压缩	压缩质量因子 1~100 对图像进行批量压缩
篡改图像内容	对图像进行旋转、添加素材、裁剪
直方图均衡化	
调节水印的叠加强度	叠加强度为 0.0001、0.001…1…100

- 不同容量水印嵌入
  - 嵌入水印大小从 64×64~114×114 像素，计算提取水印准确率及绘制 PSNR~容量曲线.
    - 嵌入的水印图像共 51 张



    - 将其按图片尺寸从小到大循环嵌入载体图像中再进行提取，准确率与水印容量的关系如下图(a)，在实验过程中模拟信道进行了加噪处理：

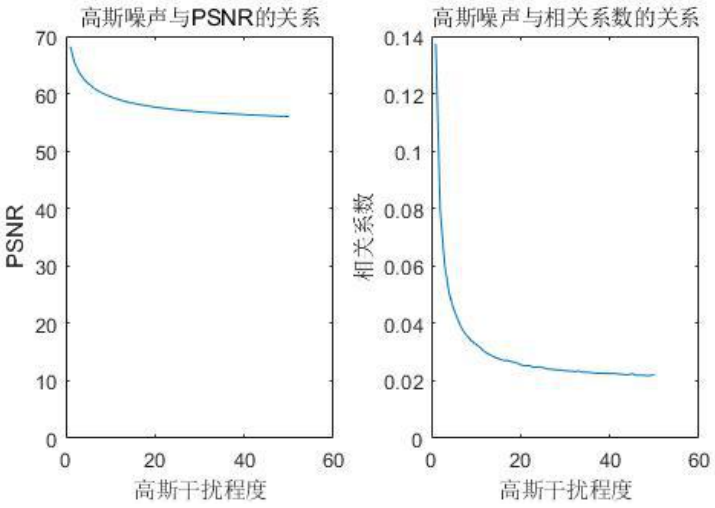


(a)水印容量与提取准确率的关系，
横轴为水印图片的空间分辨率大小

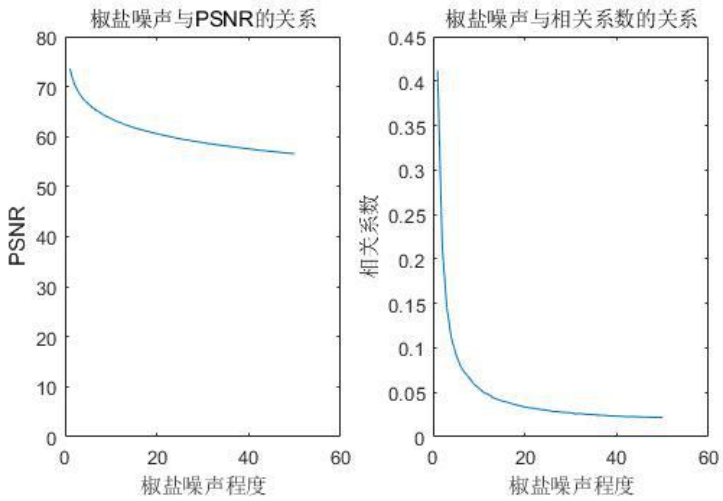
结论：由于水印嵌入容量从 64×64 开始，曲线最前端陡增.随着嵌入水印容量的增大，提取水印的准确率基本保持不变，说明水印容量对水印提取的准确性影响不大.

- 噪声攻击测试
  - 对一幅嵌入水印后的图像叠加不同程度的同一种噪声后再进行水印提取.

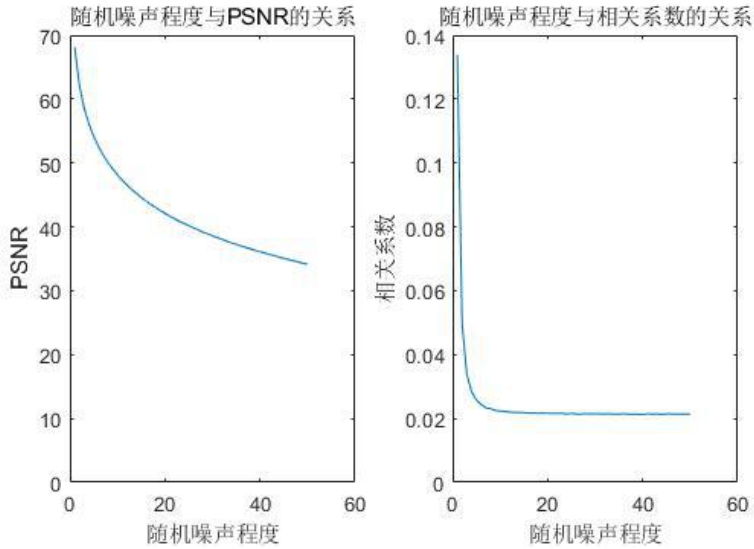
    叠加不同程度的高斯噪声，绘制其 PSNR~高斯噪声程度曲线和高斯噪声与相关系数的关系曲线.



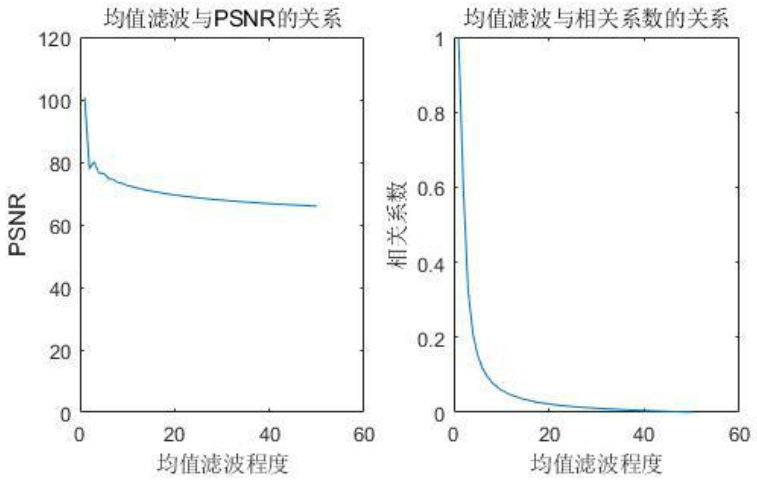
    叠加不同程度的椒盐噪声绘制其 PSNR~程度曲线和椒盐噪声与相关系数的关系曲线.



叠加不同程度的随机噪声，绘制 PSNR~随机噪声程度曲线和随机噪声程度与相关系数的关系曲线.



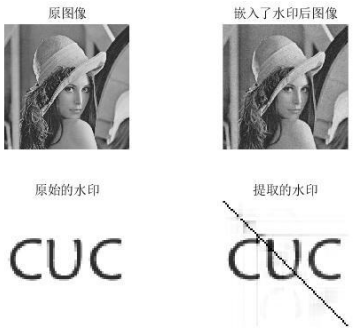
- 采用均值滤波，绘制 PSNR~均值滤波程度曲线和均值滤波程度与相关系数的关系曲线.



结论：上述几幅测试图像表明随着噪声干扰的加大，图像的 PSNR 值在减小，提取的水印与原水印的相关系数减小。其中，高斯噪声干扰、椒盐噪声干扰、均值滤波干扰随干扰程度增大其 PSNR 下降幅度逐渐减慢，干扰在增加到 10 以后基本平缓；随机噪声攻击的图像 PSNR 值持续减小；高斯噪声干扰、椒盐噪声干扰、随机噪声攻击、均值滤波攻击对于相关系数的影响为：干扰程度在 0~10 范围内时，相关系数急剧下降；达到 10 以后，相关系数基本不变。

- 对一幅嵌入水印后的图像叠加 11 种不同的噪声及滤波后再进行水印提取

高斯噪声干扰



相关系数：0.2537

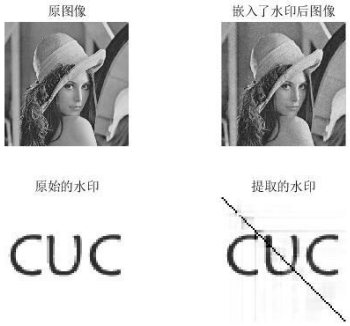
高斯噪声干扰 3×3 均值滤波

相关系数：0.3882

高斯噪声干扰 3×3 加权均值滤波

高斯噪声干扰 5×5 均值滤波

相关系数：0.5072



相关系数：0.1547

高斯噪声干扰 7×7 均值滤波

相关系数：0.0943

高斯噪声 9×9 均值滤波

相关系数：0.0667

椒盐噪声干扰

相关系数：0.2537

椒盐噪声干扰 3×3 均值滤波

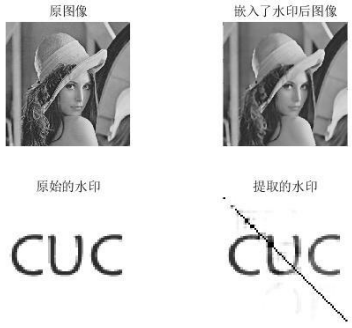
相关系数：0.1326

椒盐噪声干扰 5×5 均值滤波

椒盐噪声干扰 5×5 中值滤波

相关系数：0.0949　　相关系数：0.2798

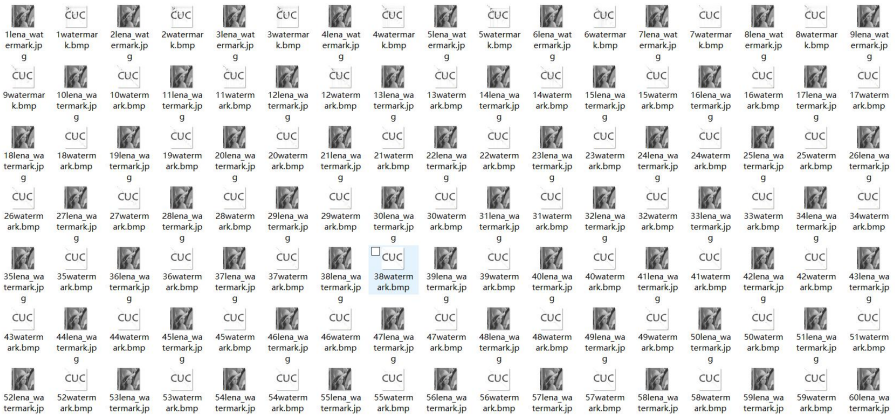
椒盐噪声干扰 7×7 中值滤波

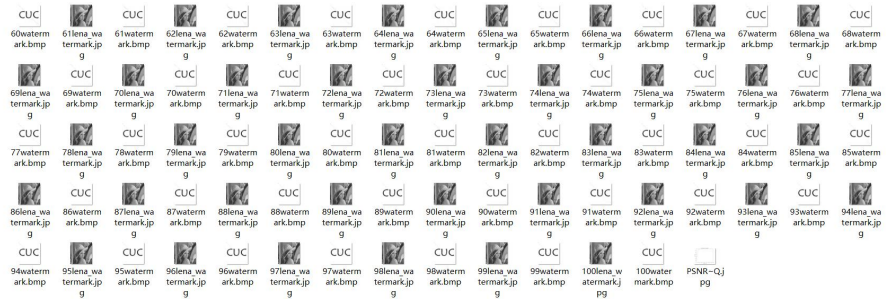


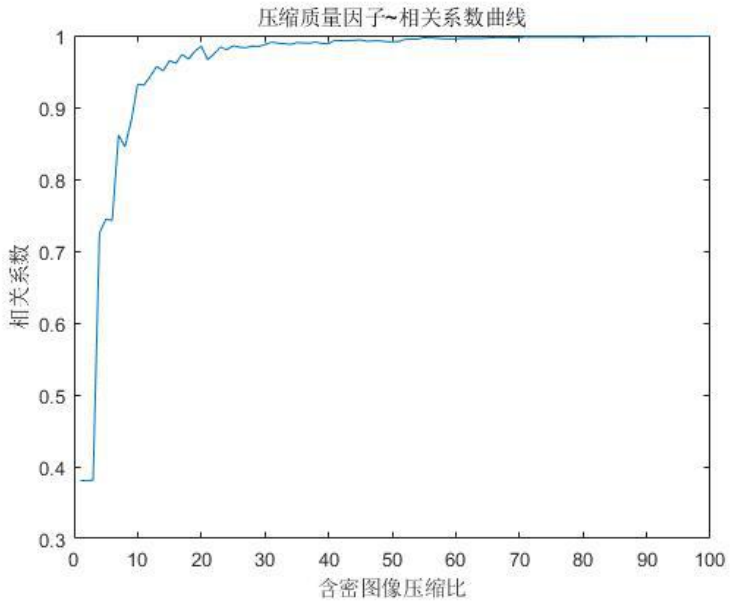
相关系数：0.1902

结论：对于不同的噪声攻击，嵌入水印后图像与提取的水印各不相同，PSNR 值与相关系数也不同。其中高　　波攻击对图像提取水印相关系数影响最大。

- 抗 JPEG 压缩测试
  - 选取压缩质量因子 Q 从 1~100 对嵌入水印的图像进行批量压缩，再依次提取，绘制 PSNR~Q 图像.







结论：随着 JPEG 压缩的质量因子不断增大，其提取水印的相关系数也不断增大。当质量因子达到 40 以上时，水印提取相关系数接近于 1.说明当 JPEG 压缩质量因子达到一定程度时，该压缩对此水印算法影响几乎为 0.

- 对图像内容进行篡改

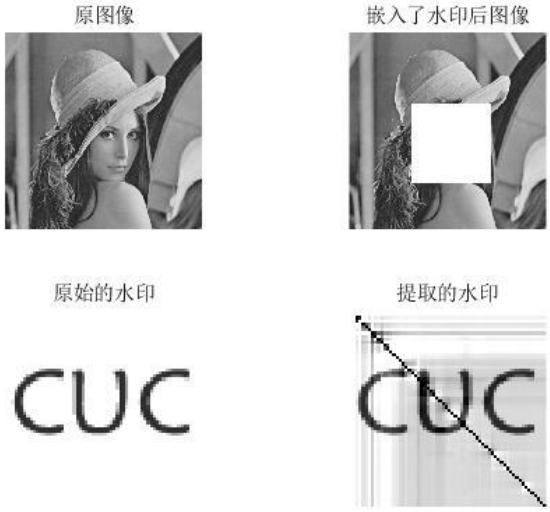
- 对嵌水印图像进行垂直翻转后进行水印提取：

相关系数：0.9988

PSNR：58.9305

结论：在进行了垂直翻转的攻击以后，从视觉上提取的水印与原始水印基本一致，其相关系数接近于 1.而图像的 PSNR 值只有 58，说明虽然图像质量下降，但对水印的提取没有很大影响，说明该水印可以抵抗几何攻击中的翻转篡改。

- 对嵌水印图像裁剪掉一部分内容后再进行水印提取：



相关系数：0.0253

PSNR：60.4801

结论：对于原图，我们进行了裁剪，虽然从视觉上看提取的水印可以看到 CUC 字样，但是相关系数只有 0.0253.说明裁剪攻击对其影响很大。

- 对嵌水印图像添加素材后再进行水印提取：

相关系数：0.0954

PSNR：66.7857

相关系数：0.1619

PSNR：71.2352

结论：在对图像进行了 ps 操作后，其对水印的影响与 ps 的范围有关，且水印的相关系数很小。说明该算法对 ps 攻击的抵抗能力较弱。

- 对含水印图像进行直方图均衡化后提取水印

相关系数：0.0564

PSNR：67.2592

结论：直方图均衡化后提取水印仍可看出 CUC 字样，其相关系数为 0.0564，说明直方图均衡化对其有一定影响。

- 对载体图像按照水印嵌入强度从大到小进行嵌入，观察其 PSNR 与相关系数.
    - 嵌入强度为 100

相关系数：1.0000

PSNR：26.9367

    - 嵌入强度为 10

原图像

嵌入了水印后图像

原始的水印

提取的水印

相关系数：1.0000

PSNR：49.8071

- 嵌入强度 1

原图像

嵌入了水印后图像

原始的水印

提取的水印

相关系数：0.9999

PSNR：73.1514

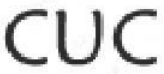
- 嵌入强度为 0.1

原图像

嵌入了水印后图像

原始的水印

提取的水印



相关系数：0.9953

PSNR：101.2910

- 嵌入强度为 0.01

相关系数：0.7037

PSNR：124.3036

- 嵌入强度为 0.001

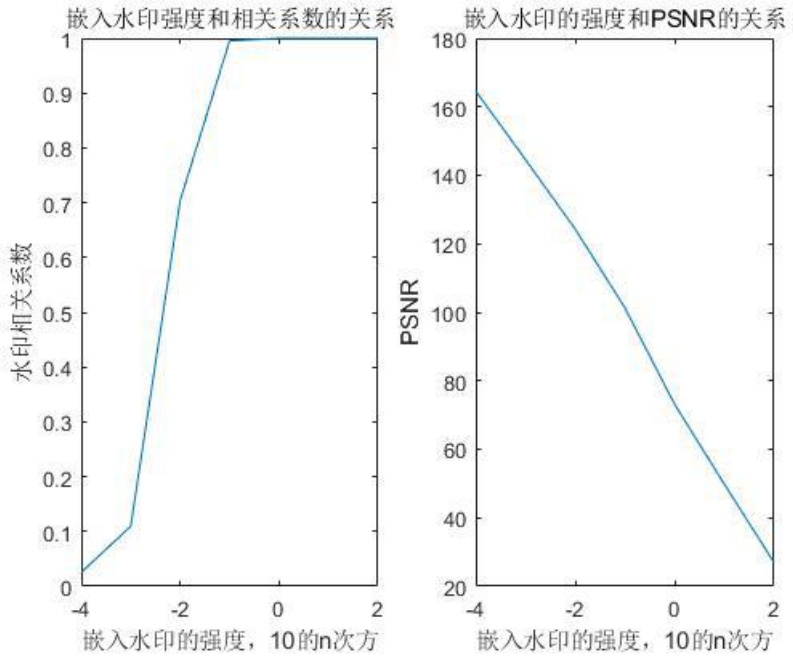
相关系数：0.1094

PSNR：144.5841

- 嵌入强度为 0.0001

相关系数：0.0256

PSNR：164.5876



结论：随着嵌入水印强度增大，水印的相似性增大，图像的 PSNR 值下降，水印的相关系数增加，当其嵌入水印强度为 1 以后，水印相关系数为 100%。

9.实验总结

- 优点：

不可见性强：水印的嵌入对图像质量影响不大.

鲁棒性强：由于矩阵的奇异值的稳定性，该算法对大部分类似于几何攻击、噪声攻击、压缩域攻击抵抗性强.

安全性高：即使非法使用者获知水印的算法，在不知道密钥（水印嵌入强度）的情况下也无法对水印进行提取和伪造.

准确性：在大部分情况下，只要获取提取水印所需要的相关信息，就能准确提取出可识别的水印.

容量大：该水印算法容量较大，在 LSB 算法中，一幅 512×512 的图像只能嵌入 64×64 的水印图像，而该算法最大可嵌入 512×512 的水印图像.

算法简单：无需对水印图像与原图像进行裁剪操作，可直接对长宽不相等且不为 8 的整数倍的图像进行水印的嵌入；直接对空域进行水印的嵌入，算法较简单，效率较高。Jsteg 算法对变换域进行操作，算法较复杂，耗时较多.

不足：

（1）无法抵抗较强的 ps 篡改
（2）嵌入强度过大时，对图像质量影响较大，有时甚至完全无法识别图像。
（3）在提取水印时，该算法需要用到关于原始载体图像的三个矩阵，所需信息较多，较为复杂。

- 总结：

本次实验主要完成了以下工作：对基于奇异值的数字水印技术研究的背景做了详细分析；给出了该数字水印的基本原理；接着就是对基于奇异值分解的数字水印技术进行了详细的分析和仿真说明，并对该水印算法做出了多方面的攻击并查看其抵御攻击的能力。最后对水印的优缺点基于本次实验进行了总结工作。

10.实验代码

- 水印的嵌入

```
clc;

clear;

close all;

I=imread('lena.bmp'); %读入原图

II=im2double(I);  %转化为[0,1)double 型  %II 为原图像

[m,n]=size(II);  %原图像大小

af=0.1;  %嵌入强度
```

```
[U,S,V]=svd(II);  %进行奇异值分解
M=imread('watermark.bmp');  %读入水印图像
W=im2double(M);  %转化为[0,1)double 型
[m1,n1]=size(W);
WW=zeros(m,n);
%进行水印信息的初始化及赋值
for i=1:m1
    for j=1:n1
            WW(i,j)=W(i,j);
    end
end
S1=S+af*WW;%加入水印后的对角阵
[U1,SS,V1]=svd(S1); %再进行奇异值分解

dlmwrite('U.txt', U1, 'delimiter', '\t','precision', 20,'newline', 'pc') %以 20 的精度存入 txt 中
dlmwrite('V.txt', V1, 'delimiter', '\t','precision', 20,'newline', 'pc')
CWI=U*SS*V';  %嵌入水印后图像
imwrite(CWI,'lena_watermark.bmp');
%fprintf('原始水印和提取水印的相关系数:%5.4f\n',nc);
subplot(1,2,1); imshow(II); title('原图像');  %显示原图像
subplot(1,2,2);  imshow(CWI); title('嵌入了水印后图像');%显示嵌入了水印后图像
```

- 水印的提取

```
clc;
clear;
close all;
U1=importdata('U.txt');%读入 U1，V1
V1=importdata('V.txt');
I=imread('lena.bmp'); %读入原图
II=im2double(I);  %转化为[0,1)double 型  %II 为原图像
[m,n]=size(II);  %原图像大小
[U,S,V]=svd(II);  %对原图进行奇异值分解
CWI=imread('lena_watermark.bmp');
CWI=im2double(CWI);
[UU,S1,VV]=svd(CWI); %对含有水印的图像进行奇异值分解
af=0.1; %强度
%NCWI=zeros(size(CWI));
SN=U1*S1*V1';  %计算中间矩阵
WN=(SN-S)/af;  %提取水印
M=imread('watermark.bmp');  %读入水印图像
W=im2double(M);  %转化为[0,1)double 型
[m1,n1]=size(W);
WNN=zeros(m1,n1);
for i=1:m1
    for j=1:n1
        WNN(i,j)=WN(i,j);
```

```
        end
    end
    subplot(2,2,3);imshow(W); title('原始的水印');
    subplot(2,2,4);imshow(WNN); title('提取的水印');
    NC=corrcoef(W,WNN);
    nc=NC(1,2);
    fprintf('原始水印和提取水印的相关系数:%5.4f\n',nc);
    subplot(2,2,1); imshow(II); title('原图像');  %显示原图像
    subplot(2,2,2);  imshow(CWI); title('嵌入了水印后图像');%显示嵌入了水印后图像
```

- 性能测试

```
%准确性---容量性能测试代码%%
clc;
clear;
close all;
k=1;
for z=1:51
      name=[num2str(k),'U.txt'];
U1=importdata(name);%读入 U1，V1
name=[num2str(k),'V.txt'];
V1=importdata(name);
I=imread('lena.bmp'); %读入原图
II=im2double(I);  %转化为[0,1)double 型  %II 为原图像
[m,n]=size(II);  %原图像大小
[U,S,V]=svd(II);  %对原图进行奇异值分解
str=num2str(k);
str=['D:\大三上\数字内容安全\project\encode\',str];
```

```
name=[str,'lena_watermark.bmp'];
CWI=imread(name);
CWI=im2double(CWI);
[UU,S1,VV]=svd(CWI); %对含有水印的图像进行奇异值分解
af=0.1; %强度
%NCWI=zeros(size(CWI));
AA=randn(size(CWI));
%[U1,SS,V1]=svd(II); %再进行奇异值分解
SN=U1*S1*V1';   %计算中间矩阵
%%加噪处理
% NCWI=CWI+AA*0.01;   %对含水印的图像加噪声
% [UU,S2,VV]=svd(NCWI); %对含有水印的图像进行奇异值分解
% SN=U2*S2*V2';   %计算中间矩阵
WN=zeros(m,n);
WN=(SN-S)/af;   %提取水印
name=['D:\大三上\数字内容安全\project\嵌入水印(1)\嵌入水印
',num2str(k+63),'.bmp'];
M=imread(name);   %读入水印图像
W=im2double(M);   %转化为[0,1)double 型
[m1,n1]=size(W);
WNN=zeros(m1,n1);
for i=1:m1
    for j=1:n1
        WNN(i,j)=WN(i,j);
    end
end
k=k+1;
% subplot(2,2,3);imshow(W); title('原始的水印');
% subplot(2,2,4);imshow(WNN); title('提取的水印');
NC=corrcoef(W,WNN);
```

```
nc(k)=NC(1,2);
str=num2str(k);
str=['D:\大三上\数字内容安全\project\decode\',str];
name=[str,'lena_watermark.bmp'];
imwrite(WNN,name);
%fprintf('原始水印和提取水印的相关系数:%5.4f\n',nc(k));

% subplot(2,2,1); imshow(II); title('原图像');   %显示原图像

% subplot(2,2,2);   imshow(CWI); title('嵌入了水印后图像');%显示嵌入了水印后

图像
end
x(1)=64;
for i =2:52
        x(i)=x(i-1)+1;
end

plot(x,nc);
title('准确性~容量曲线');

xlabel('嵌入水印图片大小');

ylabel('准确率');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



%%增加不同的噪声进行性能测试%%%%%%%%%%%%%%%%%%%%%%%%
clc
clear
close all
im=imread('lena1.bmp');
figure(1),subplot(1,3,1),imshow(im),title('原始图像')
imNois1 = imnoise(im,'gaussian',0,0.005);
figure(1),subplot(1,3,2),imshow(imNois1),title('高斯噪声干扰')

imwrite(imNois1,'高斯噪声干扰.bmp');
```

```
imNois2 = imnoise(im,'salt & pepper',0.1);
figure(1),subplot(1,3,3),imshow(imNois2),title('椒盐噪声干扰')

imwrite(imNois1,'椒盐噪声干扰.bmp');

%均值滤波后

figure(11),subplot(2,3,1),imshow(imNois1),title('高斯噪声干扰')
imNois11 = imfilter(imNois1,ones(3,3)/9);
figure(11),subplot(2,3,2),imshow(imNois11),title('3x3  均值滤波')

imwrite(imNois11,'高斯噪声干扰 3x3 均值滤波.bmp');
imNois11 = imfilter(imNois1,ones(5,5)/25);
figure(11),subplot(2,3,3),imshow(imNois11),title('5x5  均值滤波')

imwrite(imNois11,'高斯噪声干扰 5x5 均值滤波.bmp');
imNois11 = imfilter(imNois1,ones(7,7)/49);
figure(11),subplot(2,3,4),imshow(imNois11),title('7x7  均值滤波')

imwrite(imNois11,'高斯噪声干扰 7x7 均值滤波.bmp');
imNois11 = imfilter(imNois1,ones(9,9)/81);
figure(11),subplot(2,3,5),imshow(imNois11),title('9x9  均值滤波')

imwrite(imNois11,'高斯噪声干扰 9x9 均值滤波.bmp');
H=[1 2 1;2 4 2;1 2 1]/16;
imNois11 = imfilter(imNois1,H);
figure(11),subplot(2,3,6),imshow(imNois11),title('3x3  加权均值滤波')

imwrite(imNois11,'高斯噪声干扰 3x3 加权均值滤波.bmp');

%  中值滤波后

figure(22),subplot(2,3,1),imshow(imNois2),title('椒盐噪声干扰')
imNois22 = uint8(medfilt2(imNois2,[3 3]));
figure(22),subplot(2,3,2),imshow(imNois22),title('3x3  中值滤波')

imwrite(imNois22,'椒盐噪声干扰 3x3 均值滤波.bmp');
```

```
imNois22 = uint8(medfilt2(imNois2,[5 5]));
figure(22),subplot(2,3,3),imshow(imNois22),title('5x5  中值滤波')
imwrite(imNois22,'椒盐噪声干扰 5x5 中值滤波.bmp');
imNois22 = uint8(medfilt2(imNois2,[7 7]));
figure(22),subplot(2,3,4),imshow(imNois22),title('7x7  中值滤波')
imwrite(imNois22,'椒盐噪声干扰 7x7 中值滤波.bmp');
imNois22 = imfilter(imNois2,ones(3,3)/9);
figure(22),subplot(2,3,5),imshow(imNois22),title('3x3  均值滤波')
imwrite(imNois22,'椒盐噪声干扰 3x3 均值滤波.bmp');
imNois22 = imfilter(imNois2,ones(5,5)/25);
figure(22),subplot(2,3,6),imshow(imNois22),title('5x5  均值滤波')
imwrite(imNois22,'椒盐噪声干扰 5x5 均值滤波.bmp');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%



%%直方图均衡化
% CWI=histeq(CWI);    %直方图均衡化
%%直方图均衡化

CWI=im2double(CWI);
[UU,S1,VV]=svd(CWI); %对含有水印的图像进行奇异值分解
af=0.1; %强度
%NCWI=zeros(size(CWI));
SN=U1*S1*V1';    %计算中间矩阵
```

```
WN=(SN-S)/af;    %提取水印


M=imread('watermark.bmp');    %读入水印图像

W=im2double(M);    %转化为[0,1)double 型

[m1,n1]=size(W);
WNN=zeros(m1,n1);
for i=1:m1
      for j=1:n1
            WNN(i,j)=WN(i,j);
      end
end

subplot(2,2,3);imshow(W); title('原始的水印');

subplot(2,2,4);imshow(WNN); title('提取的水印');

NC=corrcoef(W,WNN);
nc=NC(1,2);

fprintf('原始水印和提取水印的相关系数:%5.4f\n',nc);

subplot(2,2,1); imshow(II); title('原图像');    %显示原图像

subplot(2,2,2);    imshow(CWI); title('嵌入了水印后图像');%显示嵌入了水印后图

像

psnr=imPSNR(II,CWI);

disp("该图片的 PSNR 值为:");

disp(psnr);




%%高斯噪声干扰%%%%%%%%%%%%%%%%%%%%%%%%%%%

% PSNR_GAOSI=zeros(50);
% x=zeros(50);
```

```
% Relate_GAOSI=zeros(50);
% for i=1:50
% a=i/100;
% imNois1 = imnoise(CWI,'gaussian',0,a);%不同的强度
% imNois1=im2double(imNois1);
% %figure(1),subplot(1,3,2),imshow(imNois1),title('高斯噪声干扰');
% %imwrite(imNois1,'高斯噪声干扰.bmp');
% PSNR_GAOSI(i)=imPSNR(imNois1,II);
% x(i)=i;
% [Ui,Si,Vi]=svd(imNois1); %对含有水印的图像进行奇异值分解
% af=0.1; %强度
% %NCWI=zeros(size(CWI));
% AA=randn(size(imNois1));
% SN=U1*Si*V1';   %计算中间矩阵
% WN_g=(SN-S)/af;   %提取水印
% WNN_g=zeros(m1,n1);
% for k=1:m1
%         for j=1:n1
%                 WNN_g(k,j)=WN_g(k,j);
%         end
% end
% NC=corrcoef(W,WNN_g);
% Relate_GAOSI(i)=NC(1,2);%计算相关系数
%
% end
%
% figure;
% subplot(1,2,1); plot(x,PSNR_GAOSI); title('高斯噪声与 PSNR 的关系');xlabel('高斯干扰程度');ylabel('PSNR');   %显示原图像
% subplot(1,2,2);    plot(x,Relate_GAOSI); title('高斯噪声与相关系数的关系');xlabel('高斯干扰程度');ylabel('相关系数');%显示嵌入了水印后图像
% %%%%%%高斯干扰测试%%%
```

```
% %%%%均值滤波%%%%%%%%%%%%%%%%%%%%%%%%
% PSNR_LB=zeros(50);
% x=zeros(50);
% Relate_LB=zeros(50);
% for i=1:50
% a=i*i;
% imNois11 = imfilter(CWI,ones(i,i)/a);
% imNois11=im2double(imNois11);
% PSNR_LB(i)=imPSNR(imNois11,II);
% x(i)=i;
% [Uii,Sii,Vii]=svd(imNois11); %对含有水印的图像进行奇异值分解
% af=0.1; %强度
% %NCWI=zeros(size(CWI));
% AA=randn(size(imNois11));
% SNi=U1*Sii*V1';    %计算中间矩阵
% WN_l=(SNi-S)/af;    %提取水印
% WNN_l=zeros(m1,n1);
% for k=1:m1
%         for j=1:n1
%                 WNN_l(k,j)=WN_l(k,j);
%         end
% end
% NC=corrcoef(W,WNN_l);
% Relate_LB(i)=NC(1,2);%计算相关系数
%
% end
%
% figure;
% subplot(1,2,1); plot(x,PSNR_LB); title('均值滤波与 PSNR 的关系');xlabel('均值滤
波程度');ylabel('PSNR');    %显示原图像
% subplot(1,2,2);    plot(x,Relate_LB); title('均值滤波与相关系数的关系');xlabel('
```

```
均值滤波程度');ylabel('相关系数');%显示嵌入了水印后图像

% %%%%%均值滤波




% %%%%%%添加椒盐噪声

% PSNR_Z=zeros(50);
% x=zeros(50);
% Relate_ZLB=zeros(50);
% for i=1:50
%     nu=i/100;
% imNois22 = imnoise(CWI,'salt',nu);
% imNois22=im2double(imNois22);
% PSNR_Z(i)=imPSNR(imNois22,II);
% x(i)=i;

% [Uiii,Siii,Viii]=svd(imNois22); %对含有水印的图像进行奇异值分解

% af=0.1; %强度

% %NCWI=zeros(size(CWI));
% AA=randn(size(imNois22));

% SNi=U1*Siii*V1';   %计算中间矩阵

% WN_z=(SNi-S)/af;   %提取水印

% WNN_z=zeros(m1,n1);
% for k=1:m1
%         for j=1:n1
%                 WNN_z(k,j)=WN_z(k,j);
%         end
% end
% NC=corrcoef(W,WNN_z);

% Relate_ZLB(i)=NC(1,2);%计算相关系数

%
% end
%
% figure;
```

```
% subplot(1,2,1); plot(x,PSNR_Z); title('椒盐噪声与 PSNR 的关系');xlabel('椒盐噪声程度');ylabel('PSNR');    %显示原图像

% subplot(1,2,2);    plot(x,Relate_ZLB); title('椒盐噪声与相关系数的关系');xlabel('椒盐噪声程度');ylabel('相关系数');%显示嵌入了水印后图像

%%%%椒盐噪声%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% %%%随机噪声%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% AA=randn(size(CWI));
%
% % [UU,S2,VV]=svd(NCWI); %对含有水印的图像进行奇异值分解

% % SN=U2*S2*V2';    %计算中间矩阵

% PSNR_S=zeros(50);
% x=zeros(50);
% Relate_S=zeros(50);
% for i=1:50
%
% num=i/10;
% imNois4 =CWI+AA*num;
% imNois4=im2double(imNois4);
% PSNR_S(i)=imPSNR(imNois4,II);
% x(i)=i;

% [Uiii4,Siii4,Viii4]=svd(imNois4); %对含有水印的图像进行奇异值分解

% af=0.1; %强度

% %NCWI=zeros(size(CWI));
% AA=randn(size(imNois4));

% SNi=U1*Siii4*V1';    %计算中间矩阵

% WN_S=(SNi-S)/af;    %提取水印

% WNN_S=zeros(m1,n1);
% for k=1:m1
%         for j=1:n1
```

```
%                 WNN_S(k,j)=WN_S(k,j);
%         end
% end
% NC=corrcoef(W,WNN_S);
% Relate_S(i)=NC(1,2);%计算相关系数
%
% end
%
% figure;
% subplot(1,2,1); plot(x,PSNR_S); title('随机噪声程度与 PSNR 的关系');xlabel('随机噪声程度');ylabel('PSNR');   %显示原图像
% subplot(1,2,2);    plot(x,Relate_S); title('随机噪声程度与相关系数的关系');xlabel('随机噪声程度');ylabel('相关系数');%显示嵌入了水印后图像
% %%%%%%%%%%%%随机噪声%%%%%%%%%%%%%%%%%%%%
```