浙江大学

课程名称：	多媒体安全
姓　　名：	庞懿非
学　　院：	计算机科学与技术学院
专　　业：	计算机科学与技术
学　　号：	3190104534
指导教师：	黄劲
完成时间：	2022 年 4 月 24 日

实验一：E_BLIND/D_LC 系统测试

一、实验目的

1.了解 E_BLIND/D_LC 系统的基本原理，理解 False Positive 和 False Negative 的概念，掌握 Linear Correlation 的计算。

2.考察 E_BLIND/D_LC 系统检测值的分布，给出检测值的概率分布图或密度分布图。

3.测试 8-bit 灰度值截断（理解为大范围的灰度取值被限制在一个小的范围中）对系统检测准确率的影响。

二、实验内容与要求

1.实现 E_BLIND/D_LC 系统。

2.设计一张水印，嵌入强度 α = 1，使用该水印测试 E_BLIND/D_LC 系统应用于不同封面时的检测准确率，计算 False Positive/Negative Rate，并绘制如课本图 3.6 的检测值分布图或密度图。要求封面数量不少于 40 张(建议使用课程提供的标准数据集作为测试封面)。

3.设计不少于 40 张不同的水印,使用固定的嵌入强度 α = 1，选择一张黑白像素比例(即灰度值接近 0 或 255 的像素数量占总像素数量的比例)不高于 30%的封面，测试不同的水印嵌入同一封面时 E_BLIND/D_LC 系统的检测准确率，计算 False Positive/Negative Rate，并绘制如课本图 3.6 的检测值分布图或密度图。

4.选取一张黑白像素比例不低于 50%的原始封面，重复子实验 3，比较检测准确率并分析原因。

三、实验环境

1.IDE: MATLAB Eclipse（Java）

2.OS: Windows 10

四、实验过程

1. 设计水印

以 512×512 为图片分辨率，使用高斯分布函数生成等大小的水印，并且将数值按照字符串存储在 hex 文件中。由于高斯分布（正态分布）生成的数值为随机浮点数，而我们需要的水印信息必须是整数，所以对将生成的浮点数进行强制类型转换成整型。

```
Random r = new Random();
(int) (r.nextGaussian() * 1);
```

由于高斯分布的特性，我们可以人为的设置它的均值和标准差，在测试中，我首先使用了标准差为 5 等较大的数值，试图对 coverwork 造成较大的影响，但是这样同样会使得最后的线性相关值过大。所以之后实验中，我按照原实验 P71 直接使用了标准的高斯函数，即均值为 0，方差为 1 的高斯函数，写入了 *a.hex* 文件，之后读取文件可以获得 watermark 的一维数组。

2. 嵌入水印，生成加水印的图片

在嵌入水印时我们统一使用 512×512 分辨率的图片，与水印信息相匹配。在提供的数据集中选择分辨率满足要求的图片。同时，可以使用 windows 的图片工具把分辨率不满足要求的图片调整至我们想要的分辨率，尽管这可能会造成一些失真（在 rec 的图片中会发现边缘效应）。

嵌入水印的过程很简单，如果想要传输信息 1，我们就把 watermark 数组对应的位置+像素中原来的像素值构成新的像素值；如果传输信息 0，则用原来的像素值-watermark 数组对应的位置构成新的像素值。当然在这一过程中，如果新的像素值不在[0,255]的区间，我们要把值 snapping 到 0 或者 255 的有效范围里。当然，在实验中，由于提供的数据全部是灰度图片，即 RGB 三元素的值都相等，所以我们在获取灰度值时可以直接选取 RGB 中的任意一个，这点在对图像信息进行读取之后可以看出。

加水印后的图像我也生成在了 *output* 文件夹中，开头为“wr_1”的图片传输信息为 1，开头为“wr0_”的图片传输的信息为 0。

```
public static int[][] getPicArrayData(String path);
public static void adding_wr1(int [][]pixel_data, int []wr_data,int [][]new_data);
public static void createNew_wr1(int [][]pixel_data, int []wr_data, int [][]new_data, String imagePath);
public static void transformGray (String imagePath, String path, String newimageType, int [][]new_data);
```

如上所示，`*getPicArrayData*`方法去获取图片中的灰度数据，通过 2 维数组返回。

`*adding_wrl*`方法指定输出图片的路径 *imagePath*，并且调用`*createNew_wrl*`方法。`*createNew_wrl*`方法计算新的像素值存入 *new_data* 中（保证像素值都在[0-255]中），然后调用`*transformGray*`方法，`*transformGray*`方法使用 *BufferedImage* 类修改图片的 RGB 值并且在指定的路径生成新的图片。

3. 计算线性相关性 Z_lc

按照公式，线性相关性的计算本质就是点积/像素数目，如下图所示。

$$z_{lc}(\mathbf{c}, \mathbf{w_r}) = \frac{1}{N}\mathbf{c} \cdot \mathbf{w_r} = \frac{1}{N}\sum_{x,y} \mathbf{c}[x,y]\,\mathbf{w_r}[x,y],$$

具体实现上，只需要读取图片的像素数据与 watermark 的数据点积然后除以像素数即可，即将两个数组对应位置相乘求平均。

```
public static double calculation(int [][]pixel_data, int []wr_data);
```

这个方法可以被其他类调用，我在生成加水印的图片的同时就调用了这个函数并且将计算结果写入了文件中，方便后续导入 Excel 或者直接使用，当然在控制台也会输出，导入文件的数据会按照我指定的格式写在文件末尾。

4. 设计 40 张不同的水印

按照标准高斯分布曲线，生成 40 张水印，存入 *wr* 文件夹中，分别命名为“a0.hex”-“a39.hex”；此外还按照标准差为 1.21 的高斯函数随机生成了 20 张水印，命名为“b0.hex”-“b19.hex”。

5. 找到合适的图片，对 E_BLIND/D_LC 系统进行检测

我们可以通过 MATLAB 对图片的像素值进行统计分析来选取满足要求的图片。

```
i=imread('lena512.bmp');%i是彩色图像
i1=im2gray(i);%i1就是灰度图像
x=i1(:);
tabulate(x);
[f,xi]=ksdensity(x);
plot(xi,f);
```

（1） 选择黑白像素比例不高于 30%的图片

我选择了 Lena 的图片，图片名称为“lena512.bmp”，在水印测试这里，利用水印文件名称的特性，我使用`*checkManyWr*`方法并且手动修改参数来做批量的运算，并将所有结果写入“result1.txt”文件中（这里也需要手动修改），具体的操作过程和之前完

全相同。

（2） 选择黑白像素比不低于 50%的图片

我选择了“rec.bmp”，这个图片一眼看上去就是只有黑色和白色，实际分析的结果也基本如此，同样适用上述方法，将结果写入“result2.txt”中。

6.数据分析

对上述三组数据，分别使用 MATLAB 分析生成的 40×3 个数据，绘制 3 条拟合的正态分布曲线，从左到右分别代表水印信息 0，无水印，水印信息 1。根据结果选择合适的阈值并且计算 False Positive/Negative Rate。具体结果与分析见第五部分。

```
x3 = sort(x3);
[mu3,sigma3] = normfit(x3);
y3 = normpdf(x3,mu3,sigma3);
plot(x1, y1, 'r', x2, y2, 'g', x3, y3, 'b');
```

这里 x3 为一列数据，将 x1，x2，x3 三组数据都进行正态分布的均值和标准差估计，并根据估计结果绘制概率密度函数。

五、实验分析与结论

part1

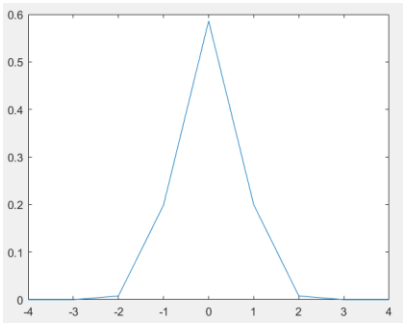
在设计水印的时候，我认为水印方差大，加水印后对原始图片的影响更大，想借此探究一下有关失真度的影响，然而，在较大的方差的情况下，无水印下的线性相关性就很强，由于误差的影响很容易出现比较大的值，所以最终我还是按照原实验要求选择了方差为 1 的水印进行了实验。

水印生成之后，我对水印的分布进行了验证。由于水印生成过程中涉及强制类型转换以及取整操作，水印必然和标准正态分布不同，即使使用`ksdensity`计算的结果仍然几乎是一条正态分布函数。

```
dat1=load('a.hex');
x=dat1(:);
tabulate(x);
x1 = sort(x);
[mu12,sigma12] = normfit(x1);
y1 = normpdf(x1,mu12,sigma12);
plot(x1, y1);
```

使用上述代码验证，结果如下：

Value	Count	Percent
-4	7	0.00%
-3	326	0.12%
-2	5431	2.07%
-1	35748	13.64%
0	179102	68.32%
1	35681	13.61%
2	5480	2.09%
3	363	0.14%
4	6	0.00%

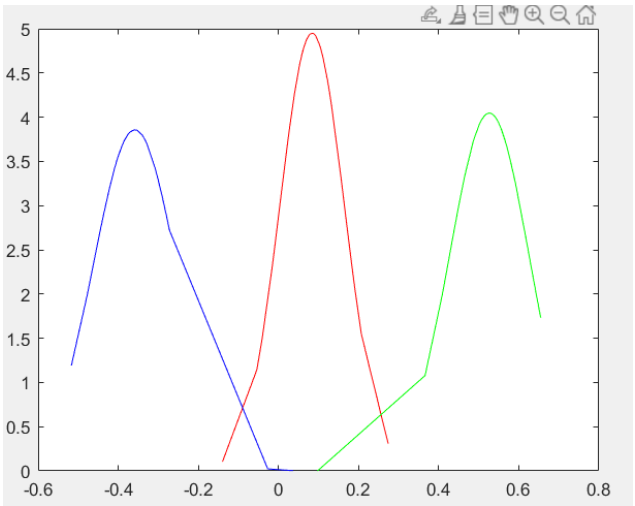


模拟出的水印分布均值为 5.264282226562500e-04，标准差为 0.680748517367049。这一部分的内容仅供参考。

在第一组嵌入水印的部分，我将所有原始图片放在 *data* 文件夹中，所有生成图片放在 *output* 文件夹中，通过比对，**原始图片、加水印信息 0 的图片、加水印信息 1** 的图片之间并没有太大的视觉差异，以 Lean 图片为例（三张图片对应关系如上），如下所示：

线性相关值分别为 0.0591087341308593 -0.404308319091796 0.522525787353515。其余图片的线性相关数据也都存储在“record.xlsx”文件中。

对上述结果进行正态分布模拟，得到如下的结果



	均值	标准差
无水印	0.084018135000000	0.080586428559647
水印信息 1	0.526401233750000	0.098581743777686
水印信息 0	-0.358419036900000	0.103414767579803

一般情况下，我们认为需要取一个阈值 τ_{lc}，来满足如下的关系。

$$m_n = \begin{cases} 1 & \text{if } z_{lc}(\mathbf{c}, \mathbf{w_r}) > \tau_{lc} \\ no\ watermark & \text{if } -\tau_{lc} \le z_{lc}(\mathbf{c}, \mathbf{w_r}) \le \tau_{lc} \\ 0 & \text{if } z_{lc}(\mathbf{c}, \mathbf{w_r}) < -\tau_{lc}. \end{cases}$$

这里我们明显可以看到，无水印信息的时候均值是不在 0 的，**所以我认为可以选取和无水印均值的一个距离作为阈值，从而更好的提高程序的自适应性**。

经过对比分析，并且结合概率分布曲线进行分析，我选择 0.35 作为距离阈值，无水印均值取 0.08。即，线性相关性低于-0.27 则认为是有水印信息 0，大于 0.43 则认为是有水印信息 1，其他情况认为无水印。

False Positive Rate：

使用 `*q=normcdf(2*mu1-0.43,mu1,sigma1)+normcdf(-0.27,mu1,sigma1)*`计算假阳性，结果为1.4391e-05。（注意，此时不可以用积分，否则会显示100%）

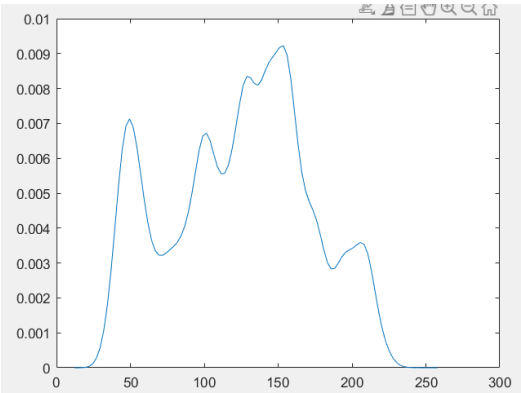
假阳性的含义是：没有水印的图片被检测出有水印，如此可见这个概率已经非常小。

False Negative Rate：

统计原结果中不在阈值内的个数。水印信息为 0 的部分中，有 2 个不小于-0.27，这两个图片都是比较特殊的，属于黑白像素比例过高的图片。水印信息为 1 的部分中，有 2 个不大于 0.43。综上，80 个水印图片中，有 4 张图片认为是无水印的，假阴性率为 5%，水印的有效性为 95%。不过，这个过程中数据量还是太小，偶然因素的印象较大，并且我还使用了高黑白像素比的图片，对结果造成了一些影响。

part2

我对图片的像素分布比例统计，选择了一张黑白像素比较低的图片以及一张黑白像素比极高的图片，如下所示：



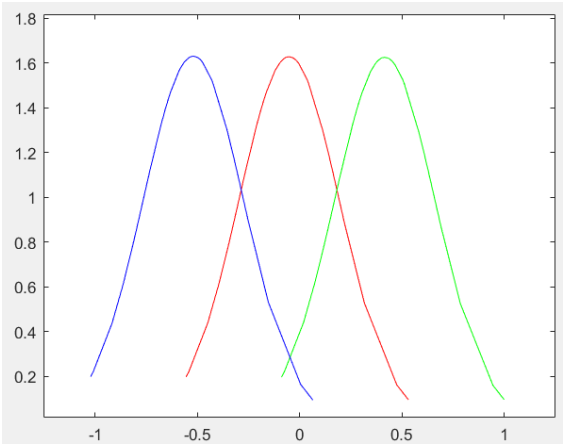
lena512.bmp

Value	Count	Percent
0	65025	24.81%
128	1020	0.39%
191	4	0.00%
255	196095	74.80%

rec.bmp

（使用的 rec.bmp 由初始的 256×256 已经调整至 512×512 大小）

在“lena512.bmp”为封面测试不同水印的结果如下，可以看到这个概率分布并不如前面的实验效果好，但是无水印的均值更加接近 0。



	均值	标准差
无水印	-0.052027511775000	0.245153669958770
水印信息 1	0.415309429200000	0.245530919902535
水印信息 0	-0.519364452400000	0.244788340280047

这里，我仍然**选取和无水印均值的一个距离作为阈值，从而更好的提高程序的自适应性**。

经过对比分析，并且结合概率分布曲线进行分析，我选择 0.41 作为距离阈值，无水印

均值取-0.05。即，线性相关性低于-0.46 则认为是有水印信息 0，大于 0.36 则认为是有水印信息 1，其他情况认为无水印。

False Positive Rate:

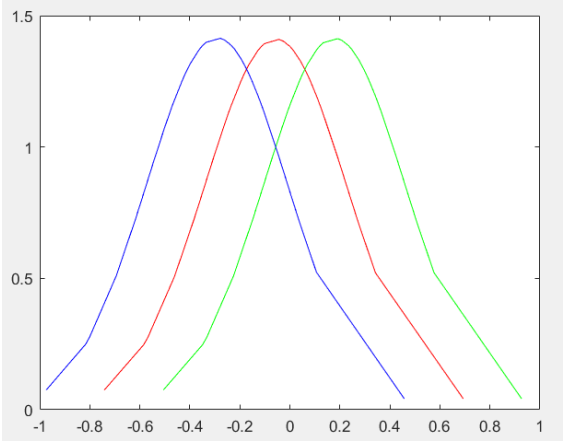
使用 `q=1-normcdf(0.36,mu1,sigma1)+normcdf(-0.46,mu1,sigma1)`计算假阳性，结果为0.0945。

假阳性的含义是：没有水印的图片被检测出有水印，这个概率相对上比较大，从标准差上也可以看出。

False Negative Rate:

统计原结果中不在阈值内的个数。水印信息为 0 的部分中，有 13 个不小于-0.46，这两个图片都是比较特殊的，属于黑白像素比例过高的图片。水印信息为 1 的部分中，有 14 个不大于 0.36。综上，80 个水印图片中，有 4 张图片认为是无水印的，假阴性率为 33.75%，水印的有效性为 66.25%。这个过程整体来说效果比较差，由于随机的水印和我选的图片不能很好的匹配，所以导致假阴性和假阳性都比较高，图中也可看出二者交叉部分很大。

在“rec.bmp”为封面测试不同水印的结果如下，可以看到这个概率分布并不如前面的实验效果好，但是无水印的均值更加接近 0。



	均值	标准差
无水印	-0.056771075400000	0.282713013246674
水印信息 1	0.177890586800000	0.282119039899995
水印信息 0	-0.291265010825000	0.281853895765122

这里，我仍然**选取和无水印均值的一个距离作为阈值，从而更好的提高程序的自适应**

性。

从之前的实验结果，我们可以知道，这组实验结果的标准差也非常大，预计假阳性和假阴性也会非常大。

经过对比分析，并且结合概率分布曲线进行分析，我选择 0.35 作为距离阈值，无水印均值取-0.06。即，线性相关性低于-0.41 则认为是有水印信息 0，大于 0.29 则认为是有水印信息 1，其他情况认为无水印

False Positive Rate：

使用 `q=1-normcdf(0.29,mu1,sigma1)+normcdf(-0.41,mu1,sigma1)`计算假阳性，结果为0.2157。

假阳性的含义是：没有水印的图片被检测出有水印，这个概率相对上很大，从标准差上也可以看出。三个曲线相邻太近，几乎无法区分。

False Negative Rate：

统计原结果中不在阈值内的个数。水印信息为 0 的部分中，有 26 个不小于-0.46，这两个图片都是比较特殊的，属于黑白像素比例过高的图片。水印信息为 1 的部分中，有 27 个不大于 0.36。综上，80 个水印图片中，有 4 张图片认为是无水印的，假阴性率为 66.25%，水印的有效性为 33.75%。这个过程整体来说效果非常差，即使将阈值设置的很小也不能避免这种情况。由于图像的黑白比例过高，导致很多水印信息无法加入到图片中，导致水印图片的区分度不足。

总结

在使用单个图片多个水印的时候，尽管有些图片看起来比较普通，黑白像素比不高，但是仍然不能对随机生成的水印有很好的适应性，这一点其实不难理解。水印的生成存在一定的偶然因素，并不是每张水印随意对一张封面进行加水印就可以得到良好的检测效果；同理，一张封面也不可能对任意的一张水印都有很好的性质。

在黑白像素比例过高的图片中加水印，很容易出现 snapping 的情况，这时水印信息其实并没有或者没有完全加到图片中，就会导致图片在加水印前后几乎没什么变化，假阳性和假阴性都会很高，这样的图片就不能使用随机生成的水印，而是需要特别的设计一个匹配的水印信息来获得比较好的效果。

六、实验感想

这个实验的原理其实特别好理解，使用的也是最简单的加水印原理，用像素值+水印值，理解和实现都不困难。但是，这个实验有趣的地方就在于它没有一个固定的答案，这实际上是一个开放性的问题，由于你选择的样本以及随机生成的水印的不同，你的实验结果可能会有很大的不同。同时，对阈值的选择也是一门学问，如何在保证假阳性率很低的情况下使得假阴性的概率也很低呢，在实现和选择的时候是一个 tradeoff。

本次实验我使用的是 Java 来完成的，在图像处理上其实 MATLAB 也可以做，但是我不太熟悉 MATLAB，使用 Java 封装的库来实现的效果也是非常好的。为了批量化处理我定义了一些规则性的文件名以及处理代码，可能会使得代码看起来有些混乱，忽略掉这些部分，整个实验的函数框架还是很简单的，主要就是分为生成水印、加水印、线性相关性计算三个类。

这个实验也是拖到现在才做，不过做出来还是非常有成就感，希望今后可以继续努力，多去实现课上学过的原理。