实验2： E_SIMPLE_8/D_SIMPLE_8系统测试

课程名称： 信息隐藏与数字水印技术

实验项目名称： E_SIMPLE_8/D_SIMPLE_8 系统测试

学生姓名： 黄炯睿 学号： 3170103455

电子邮件地址： jiongrui_huang@zju.edu.cn

实验日期： 2020 年 6 月 2 日

实验目的

1. 理解 E_SIMPLE_8/D_SIMPLE_8 系统的基本原理，掌握简单的多位信息水印技术。

实验内容与要求

1. 实现 E_SIMPLE_8/D_SIMPLE_8 系统。
2. 设计一张水印，嵌入强度 $\alpha = \sqrt{8}$，使用该水印测试 E_SIMPLE_8/ D_SIMPLE_8 系统应用于 不同封面时的检测准确率，计算 False Positive/Negative Rate 和解码的准确率。要求封面数量不 少于 40 张。False Positive/Negative Rate 的计算可以采取不同的原则。其中一种可以使用的原则 是，预先设定一个固定的阈值，8 个检测值（detect value）中有 4 个超过了阈值，就认为存在水 印，否则认为不存在水印。（也可以使用其他合理的原则，需要在报告中说明使用的是哪种原则）。准确率的计算，则是对确实添加了水印的图片，计算解码出来的信息的错误率。
3. 设计不少于 40 张不同的水印,使用固定的嵌入强度 $\alpha = \sqrt{8}$，测试E_SIMPLE_8/D_SIMPLE_8 系统应用于同一封面时的检测准确率，计算 False Positive/Negative Rate。
4. 分析信息长度增加对检测准确率的影响。

实验环境

本次实验使用MATLAB2017进行，所有脚本文件均在附件的压缩文件之中

实验过程

E_SIMPLE_8/D_SIMPLE_8 系统的实现

- E_SIMPLE_8

```
function [wimg] = E_SIMPLE_8(img, message, wi,alpha)
    wimg = zeros(size(img,1), size(img,2));
    w = (zeros(size(wi{1})));
    for i=1:size(wi,2)
        if(message(i)==1)
           w = w + wi{i};
        elseif(message(i)==0)
            w = w - wi{i};
        end
    end
    % normalize
    w_mean = mean(mean(w));
    w = w - w_mean;
```

```
    w_std = std2(w);
    w = w / w_std;

    wimg = im2double(img) + alpha*w;

    for i=1:size(img,1)
        for j=1:size(img,2)
            if wimg(i,j)>255
                wimg(i,j)=255;
            end
            if wimg(i,j)<0
                wimg(i,j)=0;
            end
        end
    end
end
```

`img` 是要嵌入进的cover image，数据类型是uint8。

`message` 是要嵌入的信息，是一个一行n列的logical矩阵，表示要嵌入信息的二进制，其中n是用户可以控制的值。

`wi` 是一个元胞数组，存放了n个高斯分布的矩阵(水印)，矩阵大小和`img`一致。

`alpha` 是嵌入强度，由用户提供，本次实验中均为$\sqrt{8}$。

`wimg` 是返回值，表示嵌入信息后的带水印的图片。

- D_SIMPLE_8

```
function [wm] = D_SIMPLE_8(wimg, wi)
    [width, height] = size(wimg);
    wm = [];
    for i=1:size(wi,2)
        w = wi{i};
        product = sum(sum(im2double(wimg).*w));
        wm(i) = double(product/(width*height));
    end
end
```

`wimg` 是一张可能带有水印的图片。

`wi` 是一个元胞数组，存放了n个高斯分布的矩阵(水印)，矩阵大小和`img`一致。

`wm` 是一张图中检测值组成的列表(1*n)。我们可以通过检测值得到信息的解码和是否含有水印等信息。

- Generate Watermark

```
function [wi] = GenerateWatermark(num,x,y,seed)
    wi = cell(1,num);
    if nargin == 3   % 如果参数里没有seed
        for i=1:num
            wi{1,i} = (randn([x,y]));  %正太分布，均值为0  方差为1
        end
    elseif nargin == 4  % 如果参数里有seed
        for i=1:num
            randn('seed',seed + i); % 可以保证同一个seed生成的水印都一样
            wi{1,i} = (randn([x,y]));  %正太分布，均值为0  方差为1
        end
    end
end
```

根据输入信息长度不同以及cover image的size不同，产生不同的水印。

实验分析与结论

问题2 使用一张水印测试不同封面时的检测准确率和解码准确率

```
clc;
path = '../data/';
list = dir(path);
seed = 0;
alpha = sqrt(8);
message =  randi([0,1],[1,8]); disp(message);
% message = [1 1 1 1 1 1 1 1];
array_wimg = [];
array_img  = [];
for i=3:(size(list,1))
    img = imread([path,list(i).name]);
    wi = GenerateWatermark(8,size(img,1),size(img,2),seed);% wi is a cell.
    wimg = E_SIMPLE_8(img,message, wi,alpha);
    array_wimg(i-2,:) = D_SIMPLE_8(wimg,wi);
    array_img(i-2,:) = D_SIMPLE_8(img,wi); % 不添加水印直接decode
end
[FalsePositive,FalseNegative ] = getDetectRate(array_wimg, array_img,0.1);
accuracy = getDecodeRate(array_wimg, message);
```

代码运行后我们比较变量`array_wimg`， `array_img`：

- array_img：

	1	2	3	4	5	6	7	8
1	5.7533e-04	0.0015	-0.0016	-2.1855e-04	-0.0015	8.9342e-04	8.0450e-04	-0.0014
2	8.6057e-04	0.0015	2.7953e-04	-6.3449e-04	-5.6302e-04	0.0012	9.2651e-04	3.5739e-04
3	6.6795e-04	8.6977e-04	-8.4170e-04	1.1561e-04	7.0133e-04	-3.2142e-04	8.3953e-05	-5.5679e-04
4	-2.6882e-04	0.0013	-5.5471e-04	-1.2449e-04	-2.1315e-04	0.0014	-1.1122e-04	-4.3691e-04
5	6.5293e-04	4.9482e-04	0.0011	6.8796e-04	-3.9758e-04	1.2926e-04	-7.6176e-04	-3.3367e-04
6	-7.3356e-05	6.2416e-04	3.1084e-04	-6.1819e-04	-3.2117e-04	-5.1503e-04	9.5011e-04	2.6550e-05
7	-4.5443e-04	1.2547e-04	9.1344e-06	4.0747e-04	3.6375e-04	1.0947e-04	3.5607e-04	-7.2039e-04
8	-8.7413e-04	-0.0027	4.9753e-04	-5.0254e-04	7.7104e-04	-0.0014	0.0011	4.6743e-04
9	-5.3773e-04	6.6781e-04	5.9727e-04	-6.7485e-04	-1.7587e-04	-0.0027	9.8339e-04	-5.8534e-04
10	-2.2044e-05	9.0032e-04	-6.6368e-05	0.0024	4.9853e-04	4.5030e-04	0.0014	5.9781e-04
11	0.0012	-4.9684e-04	-0.0011	-0.0014	6.5035e-04	1.5772e-04	-2.1859e-04	9.6129e-04
12	0.0025	4.3061e-04	9.1321e-04	-1.1547e-04	-0.0022	2.5993e-04	0.0027	6.9424e-04
13	7.4305e-04	-0.0013	0.0020	-3.9161e-04	9.6460e-04	-0.0012	5.7239e-04	-2.6642e-04
14	2.7859e-04	-3.4533e-04	0.0019	0.0010	-0.0014	-7.5473e-04	-0.0016	-0.0013
15	2.1207e-04	-8.3860e-04	0.0013	-2.7066e-04	-7.6768e-04	-0.0016	-4.0059e-05	-9.2611e-06
16	0.0021	0.0017	-0.0022	0.0036	0.0052	5.0334e-04	-0.0025	-0.0013
17	2.1827e-04	-0.0011	-6.2336e-04	-7.1414e-04	-2.4982e-04	0.0013	3.4587e-05	2.6001e-04
18	8.0661e-04	0.0010	-8.9055e-04	0.0015	-0.0020	-5.7807e-04	0.0012	-0.0011
19	-3.0312e-04	0.0010	-4.7112e-04	-1.5258e-04	-4.6159e-04	3.6025e-04	-8.0565e-04	-4.6658e-04
20	-0.0014	-0.0026	-2.8672e-04	-0.0019	3.0059e-05	0.0015	0.0033	0.0022

- array_wimg

	1	2	3	4	5	6	7	8
1	-0.6048	0.6025	0.6049	-0.5964	-0.5991	-0.5948	-0.5930	0.6018
2	-0.5310	0.5406	0.5357	-0.5386	-0.5348	-0.5312	-0.5277	0.5435
3	-0.5241	0.5354	0.5352	-0.5413	-0.5317	-0.5387	-0.5373	0.5359
4	-0.5571	0.5635	0.5508	-0.5533	-0.5535	-0.5536	-0.5607	0.5521
5	-0.5601	0.5634	0.5633	-0.5600	-0.5692	-0.5596	-0.5681	0.5583
6	-0.5528	0.5567	0.5507	-0.5517	-0.5524	-0.5523	-0.5461	0.5511
7	-0.5451	0.5462	0.5395	-0.5443	-0.5410	-0.5399	-0.5412	0.5400
8	-0.5987	0.5974	0.5999	-0.6006	-0.5894	-0.6021	-0.6005	0.5975
9	-0.5732	0.5664	0.5648	-0.5730	-0.5735	-0.5796	-0.5711	0.5705
10	-0.5625	0.5709	0.5536	-0.5520	-0.5553	-0.5655	-0.5641	0.5667
11	-0.5604	0.5596	0.5585	-0.5677	-0.5638	-0.5572	-0.5629	0.5684
12	-0.5793	0.5876	0.5905	-0.5919	-0.5916	-0.5914	-0.5760	0.5873
13	-0.5711	0.5711	0.5789	-0.5691	-0.5702	-0.5735	-0.5698	0.5656
14	-0.5197	0.5276	0.5108	-0.5099	-0.5402	-0.5167	-0.5321	0.5113
15	-0.5617	0.5604	0.5679	-0.5637	-0.5646	-0.5670	-0.5581	0.5649
16	-0.5922	0.5867	0.5815	-0.5908	-0.5818	-0.5855	-0.6046	0.5871
17	-0.5378	0.5411	0.5495	-0.5484	-0.5432	-0.5416	-0.5449	0.5472
18	-0.5632	0.5698	0.5674	-0.5689	-0.5735	-0.5695	-0.5573	0.5639
19	-0.5511	0.5592	0.5528	-0.5579	-0.5567	-0.5556	-0.5561	0.5530
20	-0.5837	0.5774	0.5975	-0.5796	-0.5994	-0.5975	-0.5828	0.6042

我们可以发现有水印和无水印的数值差距很明显，因此我们设置0.1作为阈值，采用实验报告中的原则：如果超过阈值的值的个数大于当前信息长度的一半，则认为存在水印，否则就不存在。

因此我用 `getDetectRate` 来获得False Positive/Negative Rate 的值。

```
function [FalsePositive,FalseNegative] = getDetectRate(array_wimg,
array_img, threhold)
FalsePositive = 0;
FalseNegative = 0;
for i=1:size(array_wimg,1)
    count = 0;
    for j=1:size(array_wimg,2)
        if abs(array_wimg(i,j)) < threhold
            count = count + 1;
        end
    end
    if count >= size(array_wimg,2)/2
```

```
        FalseNegative = FalseNegative + 1;
    end
end

for i=1:size(array_img,1)
    count = 0;
    for j=1:size(array_img,2)
        if abs(array_img(i,j)) < threhold
            count = count + 1;
        end
    end
    if count < size(array_img,2)/2
        FalsePositive = FalsePositive + 1;
    end
end
% FalsePositive = FalsePositive/size(array_img,1);
% FalseNegative = FalseNegative/size(array_wimg,1);
end
```

最后我们得到在该系统下，我们False Positive Rate 和 False Negative Rate 均为0.

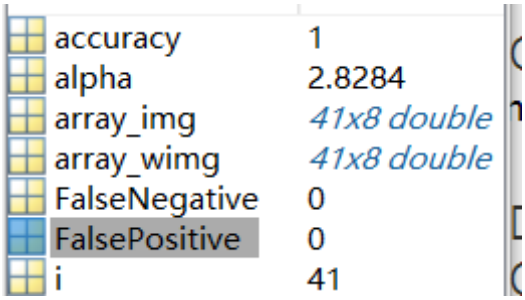
之后再来看解码的准确率。因为解码的问题只考虑有水印的图片，因此我们把有水印的函数放入函数，与嵌入的原始信息进行比较。我们发现decode的成功率为100%。

问题3 多张不同水印嵌入同一封面时的检测准确率和解码准确率

```
clc;
alpha = sqrt(8);
message = randi([0,1],[1,8]);  disp(message);
array_wimg = [];
array_img = [];
img = imread('../data/cman.BMP');
for i=1:41
    wi = GenerateWatermark(8,size(img,1),size(img,2)); % wi is a cell.
    wimg = E_SIMPLE_8(img,message, wi,alpha);
    array_wimg(i,:) = D_SIMPLE_8(wimg,wi);
    array_img(i,:) = D_SIMPLE_8(img,wi); % 不添加水印直接decode
end
[FalsePositive,FalseNegative ] = getDetectRate(array_wimg, array_img,0.1);
accuracy = getDecodeRate(array_wimg, message);
```

我们固定这次要嵌入的图片是cman.BMP, 且在调用`GenerateWatermark()`时不给seed参数，保证每次生成水印的不同。

最后的生成结果和上一个问题一样，都是完全的成功。



问题4 分析信息长度增加对检测准确率的影响。

```
clc;
alpha = sqrt(8);
img = imread('../data/rec.bmp');
for i=4:8:64
    array_wimg = [];
    array_img = [];
    message = randi([0,1],[1,i]);
    fprintf('length of message = %d\n', i);
    for j=1:41
        wi = GenerateWatermark(i,size(img,1),size(img,2)); % wi is a cell.
        wimg = E_SIMPLE_8(img,message, wi,alpha);
        array_wimg(end+1,:) = D_SIMPLE_8(wimg,wi);
        array_img(end+1,:) = D_SIMPLE_8(img,wi); % 不添加水印直接decode
    end
    [FalsePositive,FalseNegative] = getDetectRate(array_wimg,
array_img,0.1);
    accuracy = getDecodeRate(array_wimg, message);
    fprintf('FalsePositive number = %d/41 \t FalseNegative number = %d/41
\n',FalsePositive,FalseNegative );
    fprintf('Decode Accuracy = %2.2f%%\n--------------------------------
\n\n',accuracy );

end
```

我这次也是固定了图片，用问题三的方法改变水印来获得不同的准确率。对某一个固定的信息长度采用41个不同的水印，检测其中的准确率和检测率，此时的检测值的阈值设为0.1。

我从将信息的长度从8到128进行增长，每次输出当前的准确率和检测率。

```
length of message = 104
FalsePositive number = 0/41      FalseNegative number = 0/41
Decode Accuracy = 1.00%
--------------------------------

length of message = 112
FalsePositive number = 0/41      FalseNegative number = 0/41
Decode Accuracy = 1.00%
--------------------------------

length of message = 120
FalsePositive number = 0/41      FalseNegative number = 0/41
Decode Accuracy = 1.00%
--------------------------------

length of message = 128
FalsePositive number = 0/41      FalseNegative number = 0/41
Decode Accuracy = 1.00%
--------------------------------
```

可以观察到随着信息的变长，水印的嵌入和解码变得越来越耗时。但是准确率和检测率依然是百分百。

于是我进一步加大信息的长度。当信息的长度达到500时，结果如下：

```
length of message = 500
FalsePositive number = 0/41       FalseNegative number = 41/41
Decode Accuracy = 1.00%
--------------------------------
```

发现这时的False Negative Rate 非常高。查看当前的有水印图片检测值和无水印的比较：

- 有水印图片检测值

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	1
1	0.0877	-0.0703	0.0813	-0.0940	-0.0841	-0.0777	0.0750	-0.0936	-0.0682	0.0758	-0.0664	-0.0743	0.0727	0.0736	0
2	0.0783	-0.0547	0.0746	-0.0825	-0.0955	-0.0868	0.0715	-0.0551	-0.0805	0.0831	-0.0688	-0.0728	0.0801	0.0651	0
3	0.0648	-0.0827	0.0778	-0.0743	-0.0665	-0.0587	0.0957	-0.0849	-0.0794	0.0902	-0.0707	-0.0869	0.0849	0.0983	0
4	0.0686	-0.0784	0.0939	-0.0750	-0.0892	-0.0773	0.0720	-0.0522	-0.0746	0.0716	-0.0672	-0.0650	0.0879	0.0759	0
5	0.0855	-0.0816	0.0817	-0.0884	-0.0670	-0.0762	0.0765	-0.0775	-0.0823	0.0899	-0.0816	-0.0632	0.0753	0.0818	0
6	0.0898	-0.0835	0.0581	-0.0673	-0.0764	-0.0668	0.0710	-0.0753	-0.0778	0.0662	-0.0790	-0.0770	0.0755	0.0436	0
7	0.0722	-0.1014	0.0786	-0.0850	-0.0714	-0.0781	0.0726	-0.0717	-0.0830	0.0711	-0.0793	-0.0712	0.0852	0.0877	0
8	0.0745	-0.0724	0.0867	-0.0745	-0.0764	-0.0667	0.0703	-0.0817	-0.0713	0.0699	-0.0681	-0.0806	0.0679	0.0747	0
9	0.0701	-0.0789	0.0812	-0.0872	-0.0733	-0.0789	0.0778	-0.0738	-0.0681	0.0790	-0.0638	-0.0544	0.0732	0.0724	0
10	0.0824	-0.0848	0.0610	-0.0622	-0.0772	-0.0737	0.0604	-0.0558	-0.0707	0.0747	-0.0743	-0.0829	0.0823	0.0769	0
11	0.0551	-0.0923	0.0935	-0.0745	-0.0665	-0.0950	0.0719	-0.0825	-0.0785	0.0637	-0.0867	-0.0763	0.0821	0.0622	0
12	0.0679	-0.0877	0.0732	-0.0747	-0.0793	-0.0743	0.0818	-0.0724	-0.0632	0.0706	-0.0796	-0.0797	0.0762	0.0858	0
13	0.0735	-0.0698	0.0596	-0.0695	-0.0900	-0.0525	0.0799	-0.0793	-0.0732	0.0765	-0.0727	-0.0737	0.0845	0.0663	0
14	0.0863	-0.0784	0.0794	-0.0908	-0.0795	-0.0700	0.0847	-0.0948	-0.0689	0.0669	-0.0695	-0.0814	0.0604	0.0778	0
15	0.0715	-0.1002	0.0648	-0.0818	-0.0667	-0.0702	0.0759	-0.0879	-0.0744	0.0697	-0.0796	-0.0813	0.0881	0.0855	0
16	0.0702	-0.0615	0.0795	-0.0709	-0.0886	-0.0771	0.0894	-0.0612	-0.0885	0.0880	-0.0813	-0.0873	0.0859	0.0834	0
17	0.0831	-0.0811	0.0707	-0.0755	-0.0786	-0.0868	0.0906	-0.0697	-0.0684	0.0709	-0.1028	-0.0887	0.0693	0.0812	0
18	0.0888	-0.0843	0.0833	-0.0709	-0.0677	-0.0733	0.0938	-0.0802	-0.0793	0.0938	-0.0783	-0.0710	0.0803	0.0653	0
19	0.0760	-0.0744	0.0633	-0.0722	-0.0725	-0.0853	0.0836	-0.0703	-0.0796	0.0899	-0.0711	-0.0663	0.0877	0.0817	0
20	0.0710	-0.0761	0.0615	-0.0775	-0.0880	-0.0867	0.0683	-0.0830	-0.0802	0.0803	-0.0765	-0.0750	0.0712	0.0870	0
21	0.0840	-0.0972	0.0643	-0.0851	-0.0920	-0.0783	0.0807	-0.0841	-0.0617	0.0555	-0.0658	-0.0636	0.0712	0.0769	0

- 无水印图片检测值

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	3.3234e-05	-0.0013	-0.0025	-0.0044	0.0030	-0.0011	0.0015	-0.0035	0.0042	0.0034	0.0030	0.0012	-0.00
2	-0.0034	0.0044	-0.0035	-0.0017	-0.0058	-0.0049	0.0024	0.0064	-0.0034	0.0042	0.0034	0.0042	0.00
3	-0.0045	0.0014	-0.0058	1.6966e-04	-6.9572e-...	0.0028	0.0053	0.0012	-0.0034	0.0013	0.0066	-0.0011	0.00
4	-0.0051	-0.0011	0.0087	-0.0035	-0.0012	8.1105e-04	-5.3292e-...	0.0029	0.0014	0.0025	0.0041	0.0017	0.00
5	0.0060	5.1339e-04	9.5628e-04	-0.0018	0.0015	-8.4611e-...	-0.0020	0.0031	-0.0034	0.0015	0.0016	0.0028	-6.2421e
6	5.4789e-04	4.8566e-04	-0.0031	-0.0014	-3.1019e-...	0.0011	4.8285e-04	-2.7926e-...	-6.4982e-...	-0.0038	0.0020	0.0062	0.00
7	9.2427e-04	7.4934e-04	0.0026	-0.0022	7.5179e-04	0.0030	-7.3225e-...	-1.7190e-...	-5.1974e-...	0.0027	0.0025	-0.0025	0.00
8	-0.0034	5.4462e-04	6.9605e-04	2.0577e-04	-7.8490e-...	0.0049	-0.0038	0.0017	0.0013	5.0928e-04	0.0069	3.4331e-05	4.4168e-
9	0.0024	-0.0019	0.0029	-0.0078	0.0031	-0.0011	-0.0030	0.0042	-0.0016	0.0060	0.0028	0.0016	-0.00
10	0.0030	-0.0060	-6.0487e-...	4.2613e-04	-4.0280e-...	-0.0013	-0.0027	0.0030	0.0019	-9.5425e-...	-0.0012	-0.0018	-3.5736e
11	-0.0051	-0.0024	0.0028	0.0016	0.0026	-0.0043	-0.0028	-0.0041	-0.0062	-0.0017	7.2266e-05	0.0024	0.00
12	0.0019	-0.0047	-0.0012	8.5895e-04	0.0030	-8.7920e-...	0.0033	0.0054	0.0022	-2.9756e-...	0.0012	-0.0042	0.00
13	-2.3275e-...	8.1714e-04	0.0041	0.0055	-0.0023	0.0028	1.9505e-04	0.0012	5.4299e-04	-3.6147e-...	0.0035	8.5126e-04	0.00
14	2.6817e-04	0.0033	0.0068	0.0037	-0.0020	0.0035	0.0013	-0.0032	0.0010	-0.0025	0.0019	-0.0024	-0.00
15	-0.0034	-0.0030	-0.0018	-0.0022	0.0026	-0.0013	-0.0045	2.5107e-04	0.0014	-0.0022	-0.0032	0.0015	-0.00
16	0.0020	3.4943e-04	-0.0027	4.8545e-05	-0.0052	-0.0022	0.0046	0.0041	9.2473e-05	0.0024	0.0013	-0.0013	0.00
17	0.0058	-0.0072	9.7225e-04	-1.5960e-...	2.7413e-04	-0.0048	0.0024	-0.0029	0.0034	-0.0022	-0.0029	0.0018	-0.00
18	7.7107e-04	0.0020	0.0023	0.0031	-1.7817e-...	-2.1206e-...	0.0023	-0.0033	-8.8501e-...	0.0083	-2.0850e-...	0.0012	-0.00
19	-0.0025	0.0017	-0.0029	8.6449e-05	0.0040	-0.0058	3.7914e-04	-0.0014	-0.0018	0.0043	-0.0024	0.0016	0.00
20	-3.1500e-...	0.0065	-0.0031	0.0045	-0.0054	-0.0015	-0.0036	0.0011	0.0014	0.0015	8.3658e-05	2.0992e-04	-0.00
21	0.0023	-0.0055	9.7038e-04	9.6597e-04	-0.0017	0.0043	-0.0010	-0.0032	0.0029	-0.0036	0.0038	-2.2030e-...	-0.00

我们可以发现有水印图片检测值的绝对值大量小于0.1.但是其实和无水印图片的检测值还是有明显的差距。可以通过减小阈值来再次成功分离两者。但是我们也有理由猜想更长的信息长度会导致有水印图片检测值的变小。

产生上述现象的原因是因为随着信息长度加长，需要的水印也会增加，因此生成的正太分布的水印越来越难以做到线性无关。我们可以看以下的公式：

$$z_{lc}(c, w_{ri}) = \frac{1}{N}(c_0 * w_{ri} + w_a * w_{ri} + n * w_{ri}) = \frac{1}{N}(c_0 * w_{ri} + \alpha * \sum_{j=1}^{len} w_{mj} * w_{ri} + n * w_{ri})$$

因为水印之间不是线性无关的，因此$\alpha * \sum_{j=1}^{len} w_{mj} * w_{ri}$的值不是一个可以忽略的$\epsilon$，因此他会对检测值产生一定的影响，使检测值低于阈值导致水印不存在。

实验感想

实验是不难，但是过程中因为对于cell数据类型的size返回值理解错误，有个bug找了3小时……本来还想画个图显得好看一点。但等我找完了bug我觉好累啊。感觉这也讲讲也挺清楚的hhh

体会了E_SIMPLE_8/D_SIMPLE_8 系统在鲁棒性上的强壮。