**江苏科技大学**

**作 业 报 告**

课 程： 信息隐藏与数字水印

课 题： 二值图像信息隐藏

学 院： 计算机学院

学 号： 182210710119

组 员： 陈四贵

班 级： 1822107101

指导老师： 李会格

# 作业内容

使用一个特定图像区域中黑像素的个数来编码秘密信息；嵌入JUST并将其提取出来；要求画出灰度图。

# 算法与模型简述

理论上，应设立传输过程中可能被改变的像素百分比lumda、阈值R0、阈值R1等参数，来抵御传输中可能出现的失真，以及嵌入过程中较为突兀的不合理嵌入。但由于每一图像的像素分布一般不像Lena图像那样标准，我们也不好把握参数选取，很有可能出现图像分块后很多块都不满足嵌入条件的情况。因此，我们简化一下模型，在信息嵌入量较少的情况下，在每一块边缘修改尽可能少的像素，而不考虑健壮性、传输中可能改变的像素，以及每块黑白像素点，直接在每块最后一个像素点嵌入需修改的像素即可，这样也能在简化模型的情况下尽可能保证嵌入后图像改动较小。

1、二值图像信息处理

①读取（预处理后的）目标图片；

②将读取的图片按一定阈值转化为灰度图；

③将灰度图转为二值图像。

2、二值图像隐藏信息隐藏

①读取要隐藏的文本信息，将其转为ASCII码对应的二进制数据流；

②将二值图像矩阵分块，一行像素点为一块；

③依次在每块矩阵最后一个像素点置入要隐藏的信息。

3、二值图像信息提取

在明晓隐藏图像信息位数的前提下，依次提取相应数量的二值图像块的末尾像素点即可

# 算法分析

该算法较为简便，可以比较直观地嵌入我们所隐藏的信息。但由于没有设置健壮性与鲁棒性相关算法，在实际操作过程中很有可能出现一些问题，包括嵌入信息太多导致图像失真大、没考虑周围像素的情况而直接嵌入导致图像较之原来改变大等问题，不能很好地适应实际问题。此外，此种嵌入信息的方式过于简便，很容易被人破解以提取隐藏信息，安全性较低。总而言之，该方法仅用于初学者理解信息隐藏的基本原理和方法，不具备太大的实际用途。

# 运行结果

1、原图像

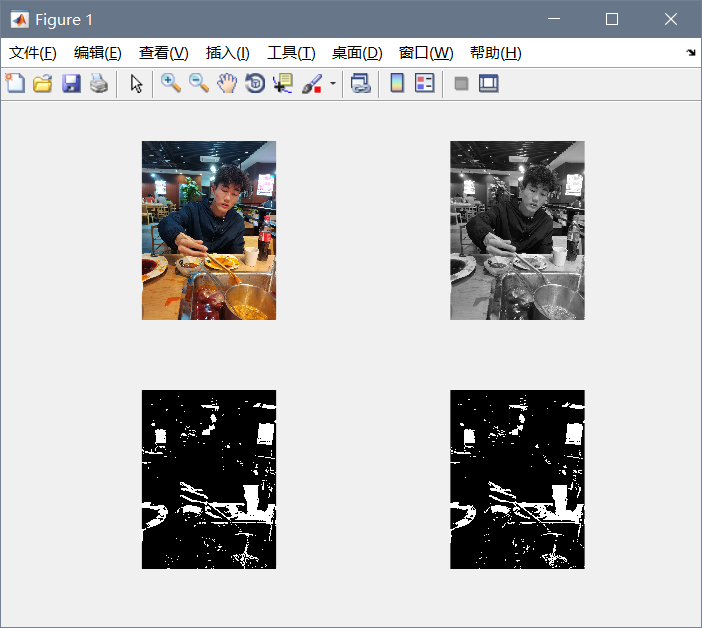
2、灰度图

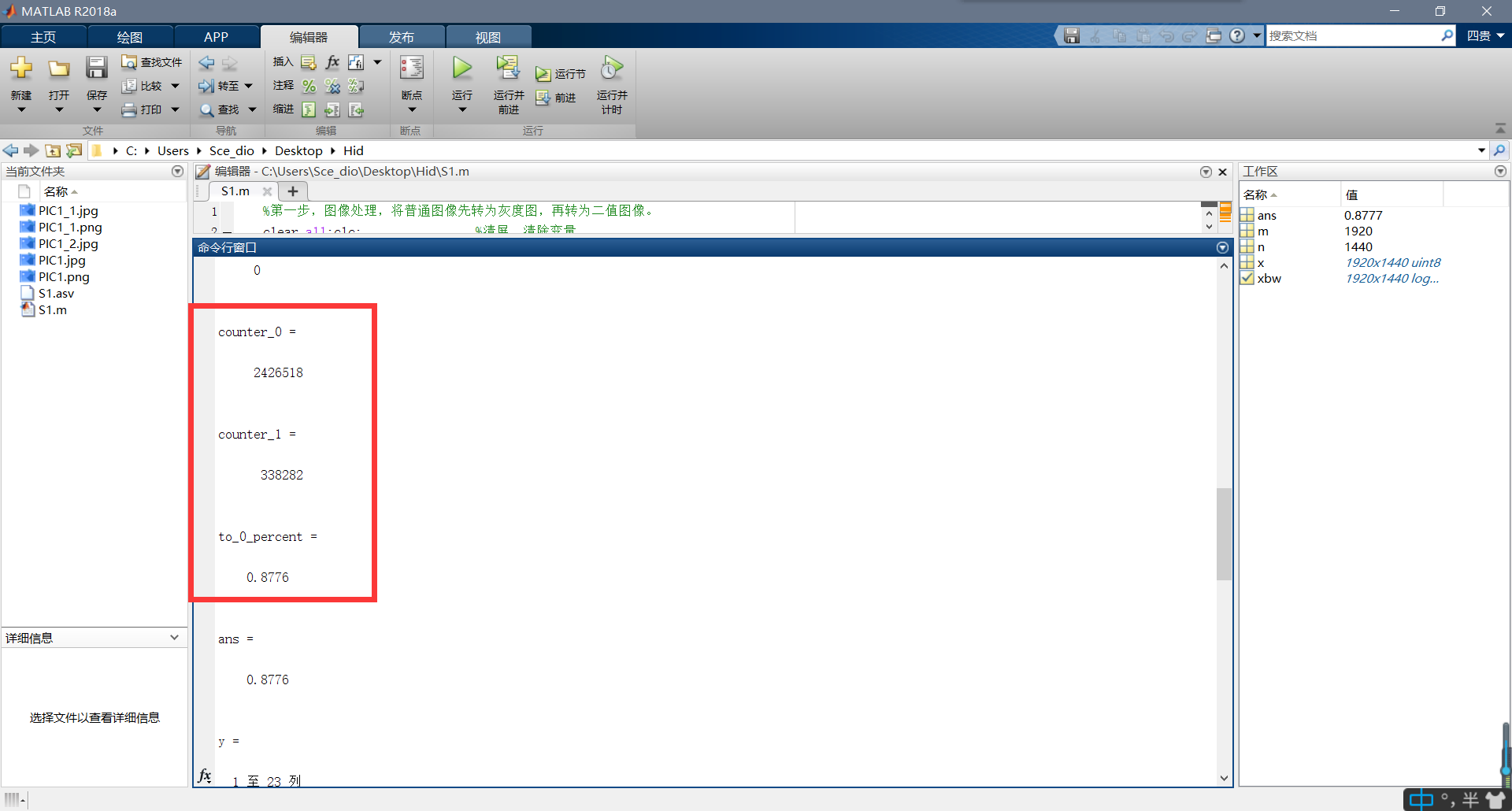
3、二值图像

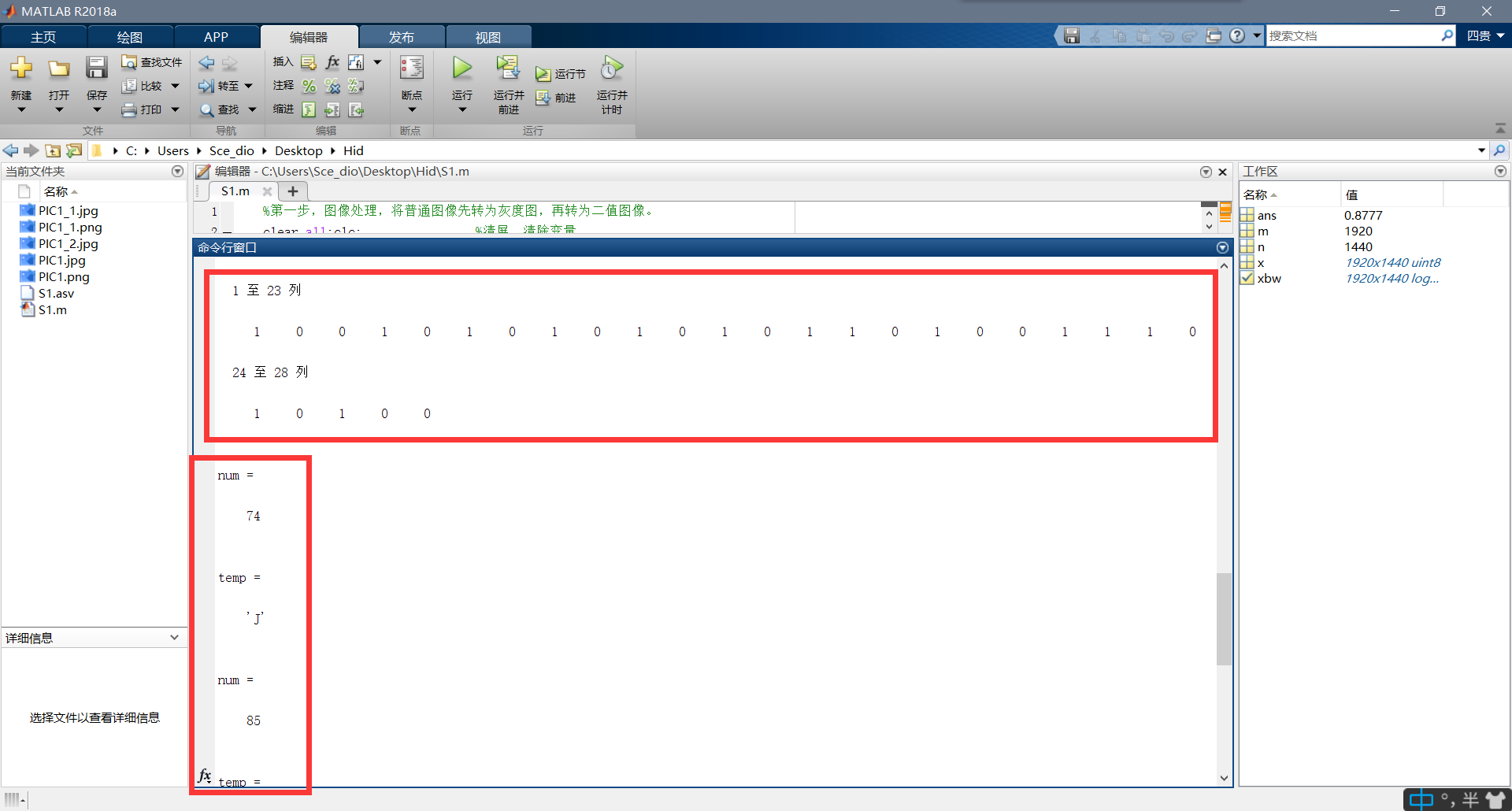
4、嵌入信息后的二值图像

5、提取信息流

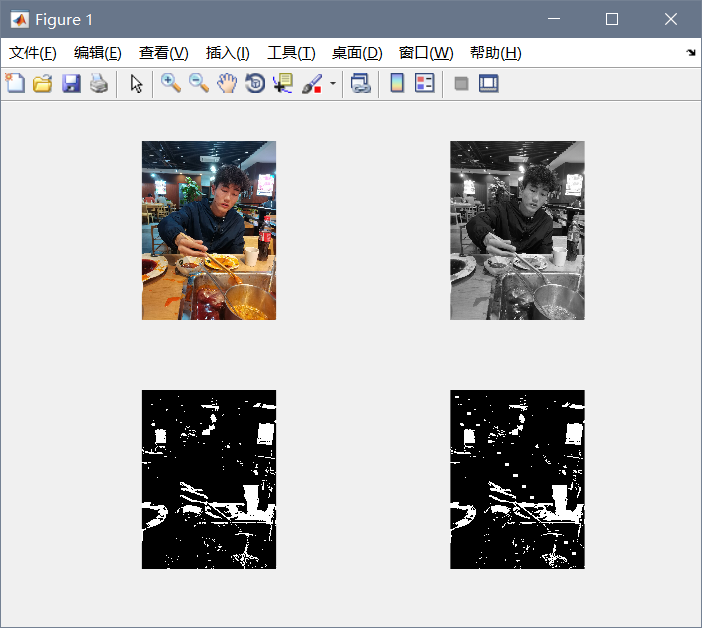
6、嵌入信息前后二值图像黑白像素点比对







# 拓展·实验教程二值图像信息隐藏代码仿真模拟结果



# 思考与体会

自拟算法是不考虑不考虑鲁棒性与图像传输中可能发生的失真，矩阵分块也不是特别科学，几乎看不出有变化。反观算法较为科学的参考嵌入算法，极为明显地看到主对角线上有若干突兀的矩形分块。究其原因，我认为主要在于在于图像像素分配的不标准，以及嵌入算法本身对图像像素比的要求较多。其次，嵌入过程中需适配嵌入的像素，这也是很重要的一个原因。通过此次实验，我认为参考算法不适用于现实生活嵌入信息，其要求过于严苛。

# 源代码·自拟算法

1. %第一步，图像处理，将普通图像先转为灰度图，再转为二值图像。
2. clear all;clc;                %清屏，清除变量
3. x=imread('PIC1.jpg');   %读取图片
4. subplot(2,2,1)     %两行两列布置中，第一张图
5. imshow(x);         %输出彩色图像
6. x=rgb2gray(x);     %将对象x进行灰度处理
7. subplot(2,2,2)     %两行两列布置中，第二张图
8. imshow(x);         %输出灰度图像
9. xbw=imbinarize(x,0.65); %再转为二值图像，二值化阈值为0.65
10. subplot(2,2,3)     %两行两列布置中，第三张图
11. imshow(xbw)        %输出黑白二值图像
12. imwrite(xbw,'PIC1\_2.jpg');    %将其二值图像保存为PIC1\_1.jpg
13. [m,n]=size(xbw)               %输出二值图像矩阵的行数m与列数n
14. print(xbw,m,n)                %Insert之前的输出
15. Insert(xbw,m,n);
17. function Insert(xbw,m,n)
18. x = abs('JUST');   %将嵌入信息转为二进制数据流
19. str='';
20. **for** i=1:4
21. str=strcat(str,dec2bin(x(i)));
22. end
23. x=str2num(str(:))
24. x=x.';                  %矩阵转置
25. j=1;
26. **for** i=1:m
27. xbw(i,n)=x(j);
28. j=j+1;
29. **if**(j==28) **break**;
30. end
31. end
32. print(xbw,m,n)
33. subplot(2,2,4);     %两行两列布置中，第四张图
34. imshow(xbw);        %输出黑白二值图像
35. Decode(xbw,m,n);
36. end
38. function Decode(xbw,m,n)
39. y=zeros(1,28);
40. **for** i=1:m
41. y(i)=xbw(i,n);
42. **if**(i==28) **break**;
43. end
44. end
45. y
46. str="";
47. **for** i=1:7:28
48. num=y(i)\*64+y(i+1)\*32+y(i+2)\*16+y(i+3)\*8+y(i+4)\*4+y(i+5)\*2+y(i+6)
49. temp=**char**(num)
50. str=strcat(str,temp);
51. end
52. str
53. end
55. function to\_0\_percent = print(xbw,m,n)
56. counter\_0=0;       %变量counter\_0用于表示黑色像素(0)的个数
57. counter\_1=0;       %变量counter\_1用于表示白色像素(1)的个数
58. **for** i=1:m          %遍历矩阵查看黑白像素点个数
59. **for** j=1:n
60. **if**(xbw(i,j)==1)
61. counter\_1=counter\_1+1;
62. end
63. **if**(xbw(i,j)==0)
64. counter\_0=counter\_0+1;
65. end
66. end
67. end
68. counter\_0          %输出黑像素点(0)的个数
69. counter\_1          %输出白像素点(1)的个数
70. to\_0\_percent=counter\_0/(counter\_0+counter\_1)     %输出黑像素点1的占比
71. end

# 源代码·实验教程算法

1. clc;
2. clear all;
3. msgfid = fopen( 'hidden.txt' , 'r');%打开秘密文件
5. [msg,count] = fread(msgfid);
6. fclose(msgfid);
7. msg = str2bit(msg);
8. msg = msg';
9. count = count \* 8;
10. io = imread('PIC1.jpg');%读入载体图像
11. subplot(2,2,1)
12. imshow( io );
13. io=rgb2gray(io);     %将对象x进行灰度处理
14. subplot(2,2,2)     %两行两列布置中，第二张图
15. imshow(io);         %输出灰度图像
16. io=imbinarize(io,0.65); %再转为二值图像，二值化阈值为0.65
17. subplot(2,2,3)     %两行两列布置中，第三张图
18. imshow(io)        %输出黑白二值图像
19. watermarklen = count;%嵌入水印信息长度,也就是载体图像分块的数量值
20. [row,col] = size(io);
21. l1=floor(row/watermarklen);%载体图像分块后的长度
22. l2= floor(col/watermarklen);%载体图像分块后的宽度
23. pixelcount = l1\* l2;%每个分块总像素的数量值
24. percent = ceil(pixelcount/2);
25. iw = io;
26. ioblack(1,watermarklen)=0;%某一个分块中黑色像素的个数
27. iowhite(1,watermarklen)=0;%某一个分块中白色像素的个数
28. n= 1;
29. **while** n<=watermarklen
30. **for** i = l1\* (n -1)+1:l1\* n
31. **for** j = l2\* (n - 1)+1:l2\* n
32. **if** io( i ,j )==0
33. ioblack(1, n)= ioblack(1 , n)+1;%计算每个分块中的黑色像素的个数
34. **else**
35. iowhite(1 ,n)= iowhite(1,n)+1;%计算每个分块中的白色像素的个数
36. end
37. end
38. end
39. n = n +1;
40. end
41. n = 1;
42. **while** n<=watermarklen
43. **if** msg(n, 1) ==1 %需要嵌入1
44. **if** ioblack(1,n)>=percent %嵌入1的时候,黑色像素比白色像素多,需要修改一些像素的颜色
45. modcount(1,n)=ioblack(1,n)-percent + 1;
46. k = 1;
47. **for** i = l1\* (n-1 )+1:l1\* n
48. **for** j = l2\* (n -1)+1:l2\* n
49. **if** (iw( i ,j) ==0&&k <= modcount(1,n))
50. iw( i , j )=1;k =k+1;
51. end
52. end
53. end
54. end
55. **else**
56. **if** iowhite(1,n) >=percent %嵌入0的时候,白色像素比罴色像素多,需要修改一些像素的颜色
57. modcount (1,n) = iowhite(1,n) - percent + 1;
58. k = 1;
59. **for** i = l1\* (n -1)+1:l1\* n
60. **for** j = l2\* (n-1)+1:l2\* n
61. **if** (iw(i , j )== 1&&k <= modcount(1,n))
62. iw( i , j)=0;k =k +1;
63. end
64. end
65. end
66. end
67. end
68. n=n + 1;
69. end
70. n=1;
72. iwblack(1 ,watermarklen)=0;%某一个分块中黑色像素的个数
73. iwwhite( 1,watermarklen)=0;%某一个分块中白色像素的个数
74. **while** n <= watermarklen
75. **for** i = l1\* (n -1)+ 1:l1\* n
76. **for** j = l2\* (n-1)+1:l2\* n
77. **if** iw( i , j) ==0
78. iwblack(1,n) = iwblack(1,n)+1;%计算每个分块中的黑色像素的个数
79. **else**
80. iwwhite(1,n)= iwwhite(1,n)+1;%计算每个分块中的白色像素的个数
81. end
82. end
83. end
84. n=n + 1;
85. end
87. imwrite(iw,'huntermarked.bmp' );
88. %figure;
89. subplot(2,2,4)
90. imshow( 'huntermarked.bmp' );