ECSE 4530 Image Processing

HW 7

Apr 2, 2018
Daniel Southwick
661542908

```
%% ==================================7.1====================================
```

Main Function

```
%% 1a
hotsauce = imread('hotsauce.png');
letter = imread('letter.png');
C = normxcorr2(letter,hotsauce);
imshow(C);
colormap jet;
```

Matlab Result

Magnitude Result as an Image

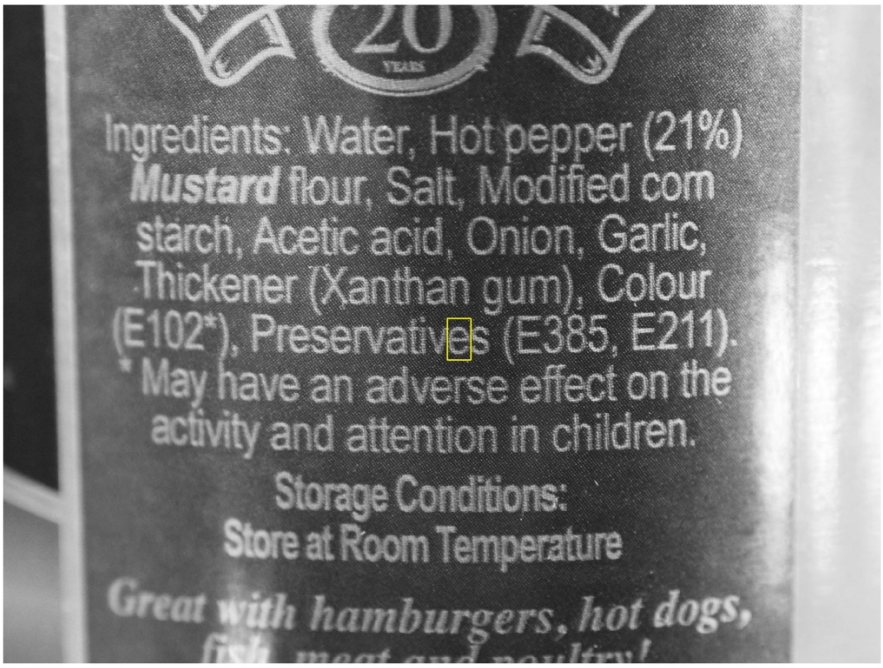
Peaks using color map Jet

The result of the normxcorr2 operation is shown above. Areas where the cross-correlation is large in magnitude is where the template is matched closely to the image. Since the template is an image of the letter “e”, the regions where the magnitude is large are locations where there is an “e” in the image. Some lower magnitude but still highlighted regions are where there are letters in general.

Main Function

```
%% 1b
[ypeak, xpeak] = find(C > 0.9);
yoffSet = ypeak-size(letter,1);
xoffSet = xpeak-size(letter,2);
imshow(hotsauce);
for i = 1:size(ypeak,1)
    rectangle('Position', [xoffSet(i)+1, yoffSet(i)+1, size(letter,2),
size(letter,1)], 'EdgeColor', 'yellow');
end
```

Matlab Result



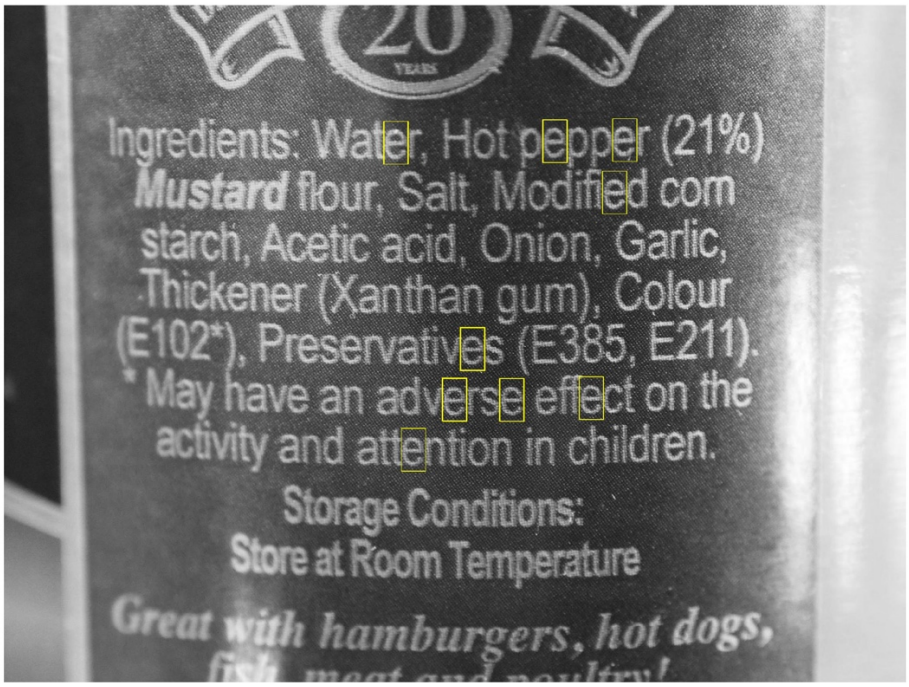
Correlation greater than 0.9

The image with cross correlation over 0.9 is shown above, it detected one single “e” in the image, since the correlation cap is high, we will only detect the place where the templates came from.

Main Function

```
%% 1c
[ypeak, xpeak] = find(C > 0.8);
yoffSet = ypeak-size(letter,1);
xoffSet = xpeak-size(letter,2);
imshow(hotsauce);
for i = 1:size(ypeak,1)
    rectangle('Position', [xoffSet(i)+1, yoffSet(i)+1, size(letter,2),
size(letter,1)], 'EdgeColor', 'yellow');
end
```

Matlab Result



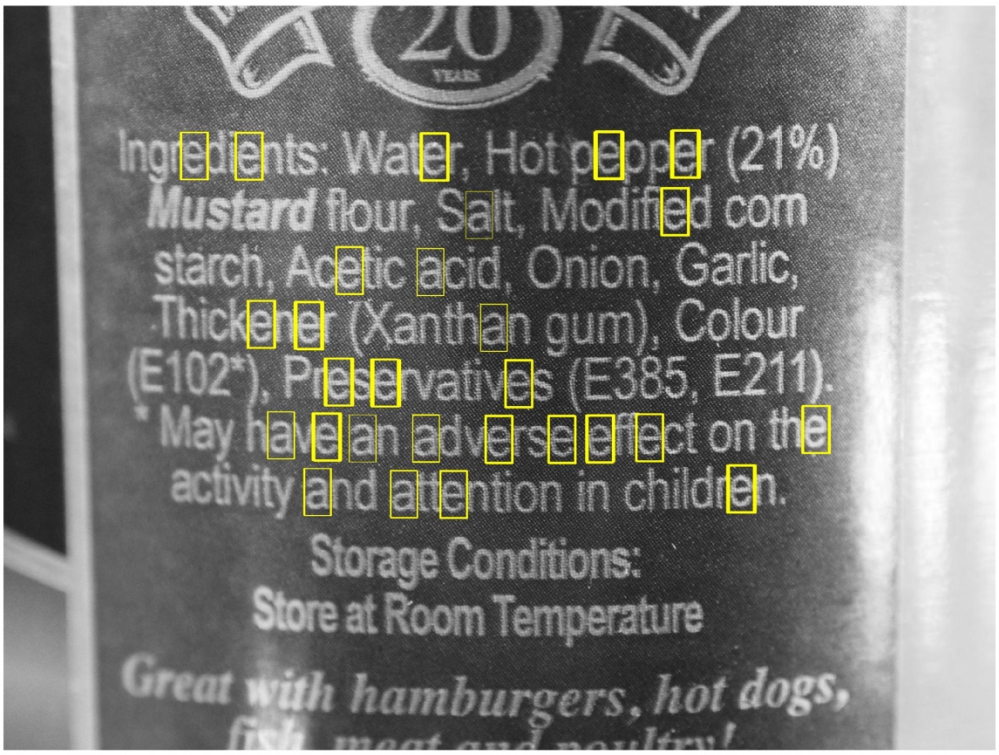
Correlation greater than 0.8

The image with cross correlation over 0.8 is shown above, it detected multiple “e” in the image, lower the cap from 0.9 to 0.8 significantly improved the detection, this operation detected nearly every “S” within the lower subtext of the image.

Main Function

```
%% 1d
[ypeak, xpeak] = find(C > 0.7);
yoffSet = ypeak-size(letter,1);
xoffSet = xpeak-size(letter,2);
imshow(hotsauce);
for i = 1:size(ypeak,1)
    rectangle('Position', [xoffSet(i)+1, yoffSet(i)+1, size(letter,2),
size(letter,1)], 'EdgeColor', 'yellow');
end
```

Matlab Result



Correlation greater than 0.7

The image with cross correlation over 0.7 is shown above, it detected multiple “e” in the image, but this is also where it begins to incorrectly identify certain peaks. With a lower threshold, more noise is brought in. Letters such as “a” was falsely marked as similar to the template.

```
%% ==================================7.2=====================================
```

Main Function

```
%% 2a
Liq1 = rgb2gray(imread('Liq1.jpg'));
Liq2 = rgb2gray(imread('Liq2.jpg'));
corners1 = detectHarrisFeatures(Liq1);
corners2 = detectHarrisFeatures(Liq2);
[features1,valid_points1] = extractFeatures(Liq1,corners1);
[features2,valid_points2] = extractFeatures(Liq2,corners2);
indexPairs = matchFeatures(features1,features2);
matchedPoints1 = valid_points1(indexPairs(:,1),:);
matchedPoints2 = valid_points2(indexPairs(:,2),:);
showMatchedFeatures(Liq1,Liq2,matchedPoints1,matchedPoints2);
```

Matlab Result

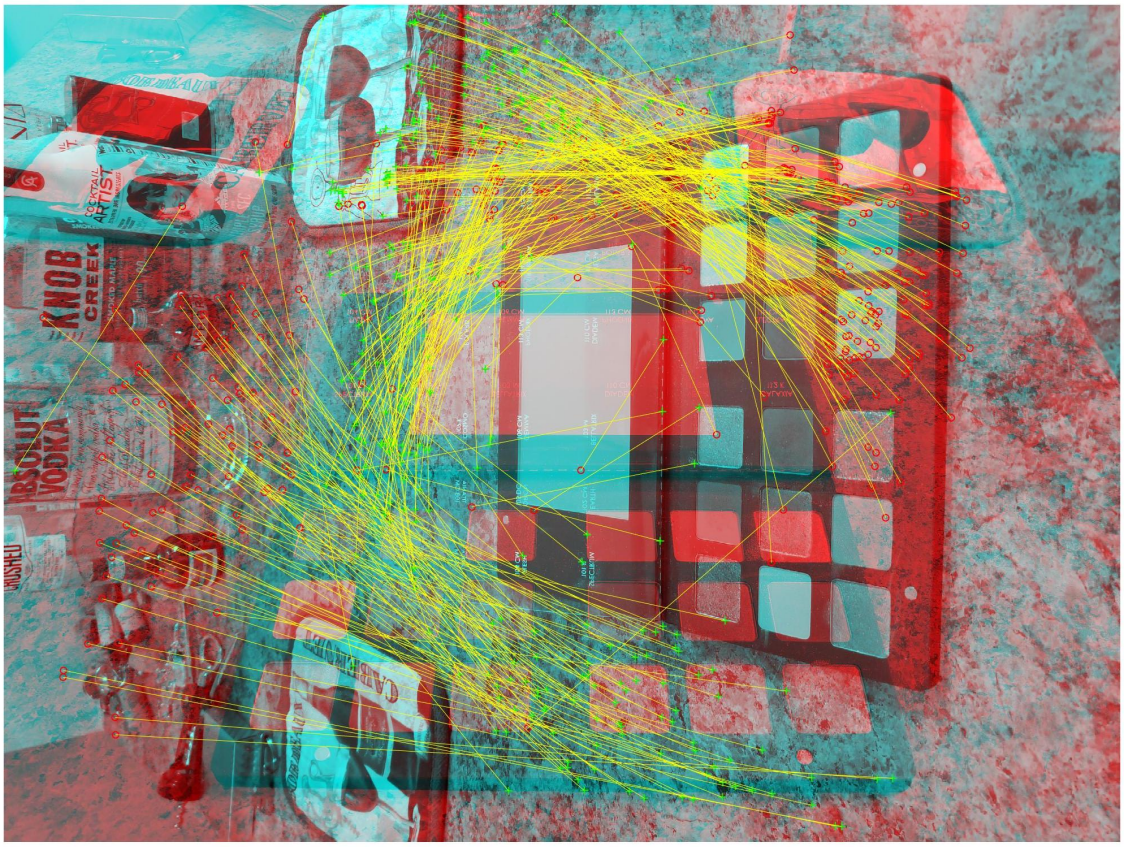
Harris Feature Parallel

Above match Features compared my girlfriend's makeup on the kitchen counter before and after a parallel shift. Matching lines are consistence as shown above, all matching lines have generally the right direction. Features mostly picked up the patterns of the counter top, pixels of boxes that have strong contrasts, the black little areas on the counter top. Matching Features are evenly distributed across the picture.

Main Function

```
%% 2b
Liq1 = rgb2gray(imread('Liq1.jpg'));
Liq2 = rgb2gray(imread('Liq3.jpg'));
corners1 = detectHarrisFeatures(Liq1);
corners2 = detectHarrisFeatures(Liq2);
[features1,valid_points1] = extractFeatures(Liq1,corners1);
[features2,valid_points2] = extractFeatures(Liq2,corners2);
indexPairs = matchFeatures(features1,features2);
matchedPoints1 = valid_points1(indexPairs(:,1),:);
matchedPoints2 = valid_points2(indexPairs(:,2),:);
showMatchedFeatures(Liq1,Liq2,matchedPoints1,matchedPoints2);
```

Matlab Result



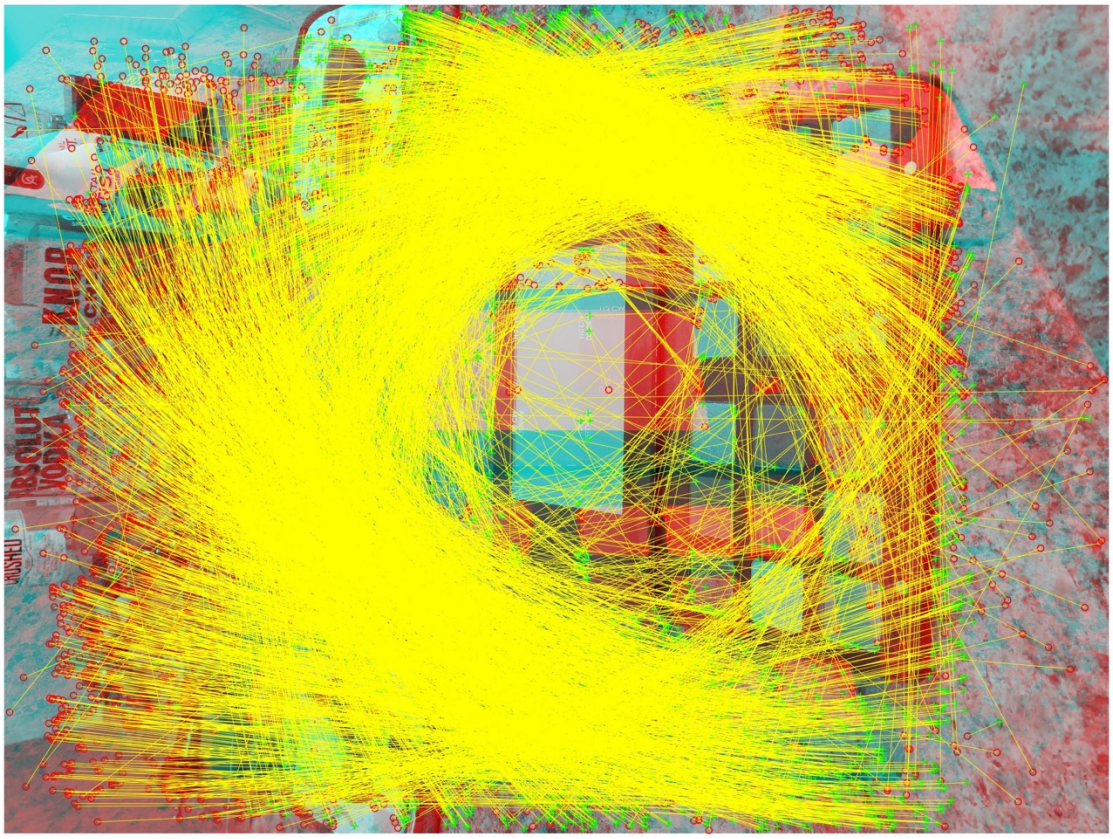
Harris Feature Rotation + Zoom In

After moving substantially from the first perspective, in the above case, we rotated the angle and zoomed it in, though the general direction can give us a sense of it's being rotated 90 degrees, some of the black dots on the countertop was mapped to the wrong places, this is shown by the matching arc that aren't reaching the full 90 degrees. The result got worse, part of the countertop got matched to the Vodka label, but the Vodka did not even exist in the first picture.

Main Function

```
%% 2c
Liq1 = rgb2gray(imread('Liq1.jpg'));
Liq2 = rgb2gray(imread('Liq3.jpg'));
corners1 = detectSURFFeatures(Liq1);
corners2 = detectSURFFeatures(Liq2);
[features1,valid_points1] = extractFeatures(Liq1,corners1);
[features2,valid_points2] = extractFeatures(Liq2,corners2);
indexPairs = matchFeatures(features1,features2);
matchedPoints1 = valid_points1(indexPairs(:,1),:);
matchedPoints2 = valid_points2(indexPairs(:,2),:);
showMatchedFeatures(Liq1,Liq2,matchedPoints1,matchedPoints2);
```

Matlab Result



SURF Feature Rotation + Zoom In

By using SURF Features, we have not only detected the corner, we also take the description of these corners into consideration. The above picture was produced following the same procedure as part b. It sure picked out a lot more good features than Harris, but also whole lot more bad matches came along the way. The whole section of liquor was picked out by SURF mistakenly as counter top black dots.

```
%% ==================================7.3=================================
```

Entropy Function

```
function ent = entropy3(X)
X=reshape(X,1,[]);
range = [double(min(X)):1:double(max(X))];
[counts,local]=hist(double(X),range);
freq = counts/sum(counts);
freq=freq(freq~=0);
ent=-sum(freq.*log2(freq));

function [E1,E2,E3,E4,E5] =  entropy3(X)
%% The entropy of the input image
E1=my_entropy(X);
pic2add=reshape(X',2,[]);
pic2freq=double(zeros(1,length(pic2add)/2));
pic2freq=double(pic2add(1,:))*256+double(pic2add(2,:))
%% The entropy of horizontally adjacent (non-overlapping)
E2= entropy3(pic2freq)/2;
pic3add=reshape(X,2,[]);
pic3freq=double(zeros(1,length(pic3add)/2));
pic3freq=double(pic3add(1,:))*256+double(pic3add(2,:));
%% The entropy of vertically adjacent (non-overlapping)
E3= entropy3(pic3freq)/2;
pic4 = reshape(X',1,[]);
pic4q=pic4(2:end);
pic4freq=double(pic4q)-double(pic4(1:length(pic4)-1));
%% The entropy of horizontally difference
E4= entropy3(pic4freq);
pic5 = reshape(X,1,[]);
pic5q=pic5(2:end);
pic5freq=double(pic5q)-double(pic5(1:length(pic5)-1));
%% The entropy of vertically difference
E5= entropy3(pic5freq);
```

```
%% ==================================7.4=====================================
```

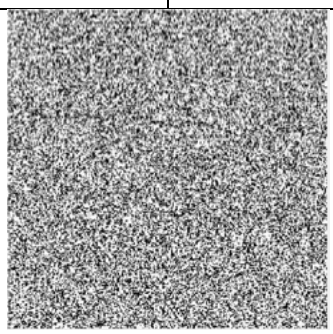
Main Function

```
%% 4
noise = imread('noise.png');
stripes = imread('stripes.png');
cells = imread('cells.png');
[En1,En2,En3,En4,En5] =  my_entropy_value(noise);
[Es1,Es2,Es3,Es4,Es5] =  my_entropy_value(stripes);
[Ec1,Ec2,Ec3,Ec4,Ec5] =  my_entropy_value(cells);
```

Matlab Result

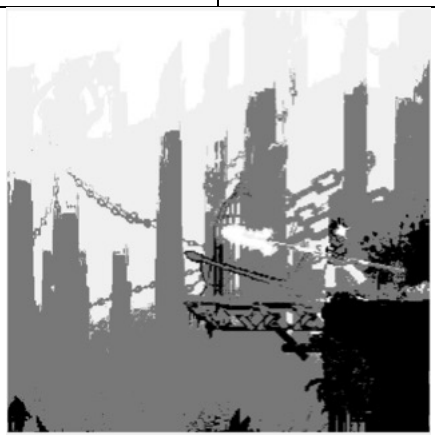
Ec1	1.7797
Ec2	1.0180
Ec3	0.9867
Ec4	0.2829
Ec5	0.2313
En1	2.0000
En2	1.9999
En3	1.9999
En4	2.6495
En5	2.6562
Es1	1.9656
Es2	1.5537
Es3	0.9828
Es4	1.5668
Es5	0.0159

Noise	
Entropy	2.0000
Horizontally Adjacent	1.9999
Vertical Adjacent	1.9999
Horizontal Difference	2.6495
Vertical Difference	2.6562



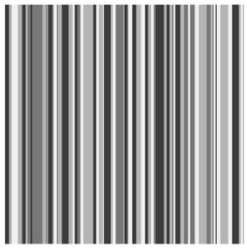
The original entropy is 2, it stayed at 2 for the adjacent pixels, and it increases to 2.65 when we take differences of adjacent pixels. The uncertainty increases because there are only 4 possible values (0 ~ 3), and when we take differences, there are 7 possible values (-3 ~ 3). There isn't any difference between vertical difference and horizontal difference because the noise is randomly distributed.

Cells	
Entropy	1.7797
Horizontally Adjacent	1.0180
Vertical Adjacent	0.9867
Horizontal Difference	0.2829
Vertical Difference	0.2313



The original entropy is ~1.8, it decreased to around 1 for the adjacent pixels, and it decreased to ~0.25 when we take differences of adjacent pixels. This is caused by the probabilities at the vertical and horizontal. Cells are mostly horizontally connected.

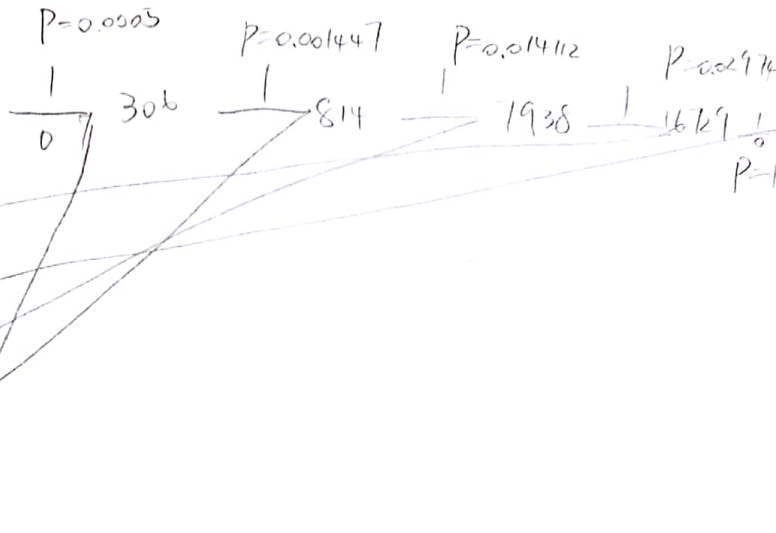
Stripes	
Entropy	1.9656
Horizontally Adjacent	1.5537
Vertical Adjacent	0.9828
Horizontal Difference	1.5668
Vertical Difference	0.0159



The original entropy is ~2, it decreased to around ~1.6 for the horizontal pixels, and it decreased to under 1 when we take differences of vertical pixels. This is caused by the probabilities when taking the connected and the difference between pixels. Since the input image is horizontal stripes, P horizontal > P Vertical and grate contrast between stripes.

7.5

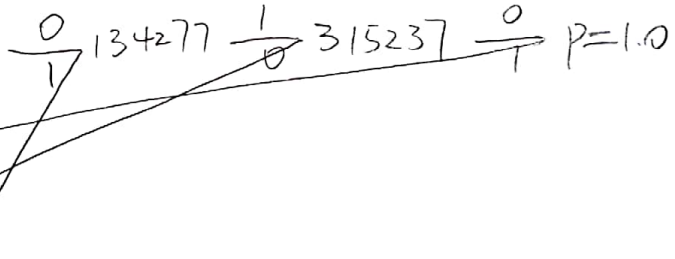
P	Vertical difference	Count
0.000096	-2	54
0.0156	-1	8791
0.9703	0	545770
0.0127	1	7124
0.0009	2	508
0.000448	3	252



H-code	length
1 1 1 1 1	5
1 0	2
0	1
1 1 0	3
1 1 1 0	4
1 1 1 1 0	5

average length = 3.333

P	Value	Count
0.01194	0	67178
0.4396	1	247264
0.3217	2	180960
0.1193	3	67098



H-code	length
0 1 0	3
1	1
0 0	2
0 1 1	3

average length = 2.25

The above result is consistance with the cells entropy result from the last question, the average length is higher on the vertical differences, caused by the vertical stripes in the picture.