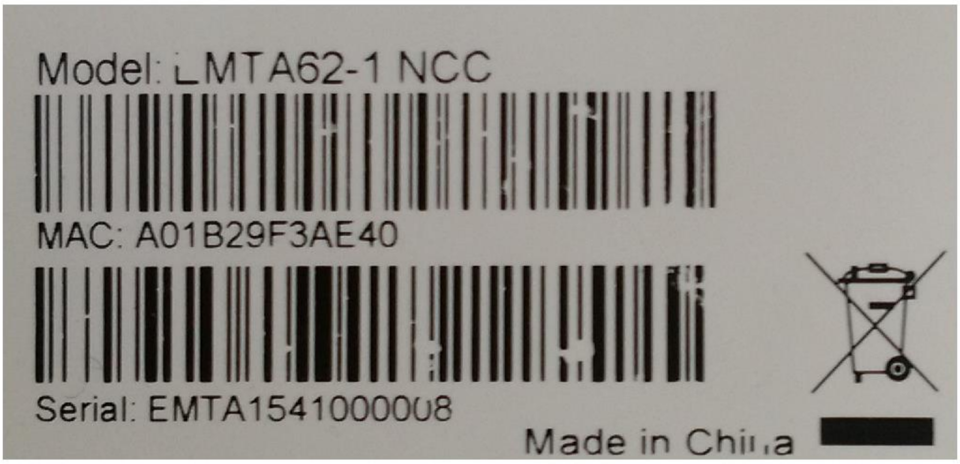
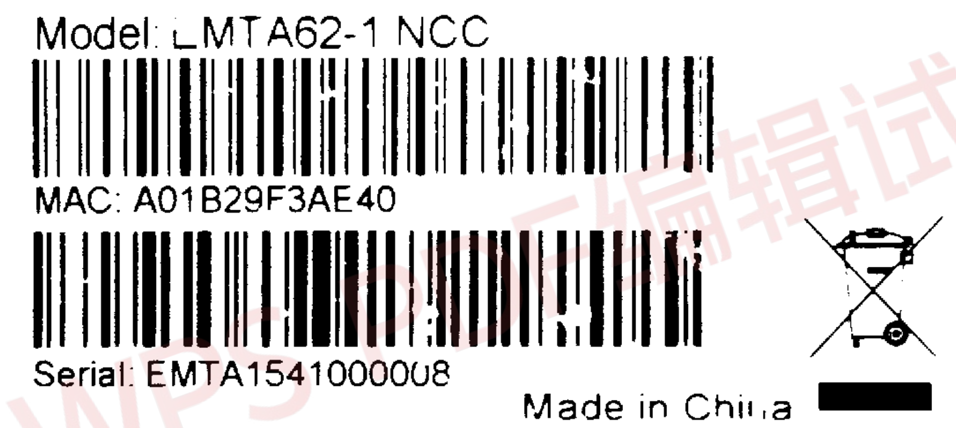
二维码瑕疵检测实验报告 [redacted]

```
% 思路：一、二维码图转化为二值图

%       二、先闭后开去除不感兴趣的字母，并将二维码的黑色条纹连成黑色方块

%       三、分别在横向和纵向对第二步得到的图像像素累加，通过检测峰值谷值确定二维码区域

%       四、原始图像中选定二维码区域 - 选定的二维码区域先开后闭 = 瑕疵点

%       五、红框标记瑕疵点


clear;close all; clc;

warning('off')
```

Binarize the input image

```
im = imread('test images/barcode_7.png');


% Origin Image

figure;imshow(im);title('Origin Image','FontSize',15);


% Gray Scale Image

im_gray = rgb2gray(im);

level = graythresh(im_gray);


% Binary Image

bw = im2bw(im_gray, level);

figure;imshow(bw);title('Binary Image','FontSize',15);
```


Origin Image


Binary Image

请基于二值图像 bw，将二维码瑕疵区域检测出来

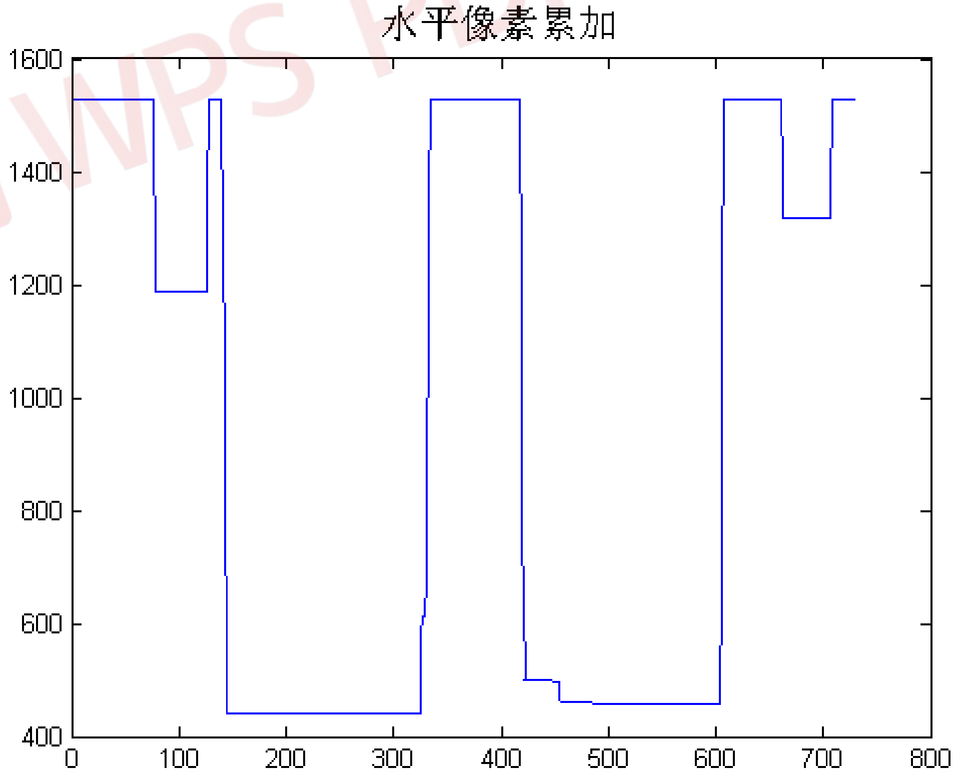
```
% Close

se = strel('rectangle',[45,1]);

fc = imclose(bw,se);

% Open

se = strel('rectangle',[3,100]);

fc2 = imopen(fc,se);

% Close

se = strel('rectangle',[2,100]);
```

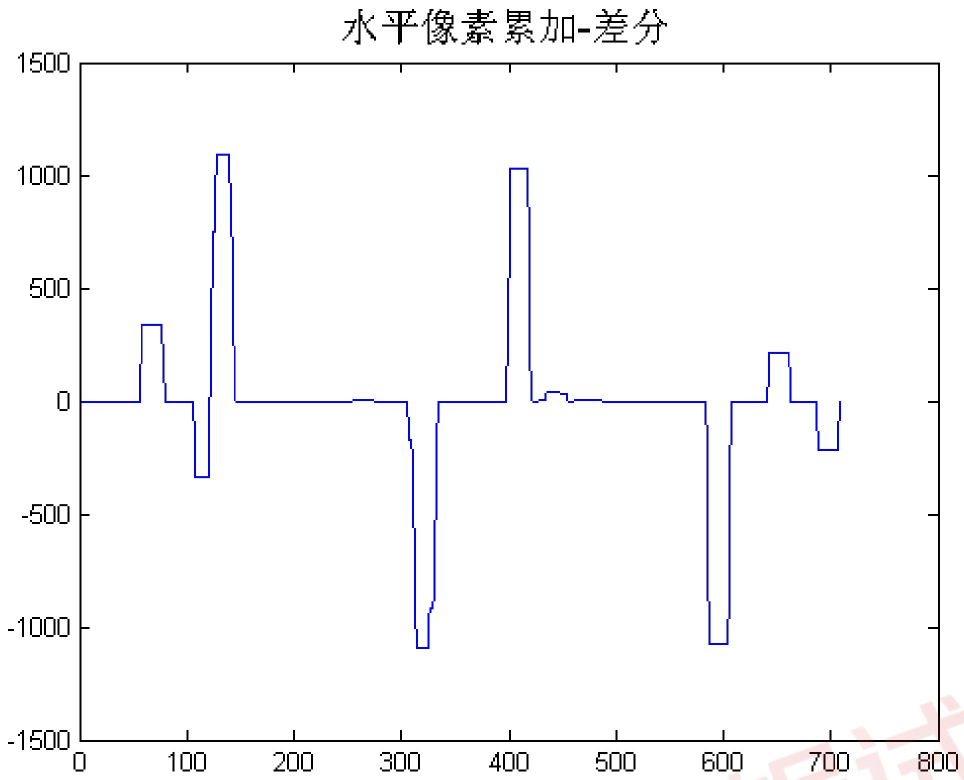
```
fc3 = imclose(fc2,se);
% Open
se = strel('rectangle',[2,50]);
fc4 = imopen(fc3,se);
% Location
sum_x = 0; sum_y = 0;
[fc4_m,fc4_n]=size(fc4);
for i=1:1:fc4_m
sum_data_x(i) = sum(fc4(i,:));
end
```

Vertical Range

```
% Determining the  range from the peak and valley values
figure;
plot([1:1:fc4_m],sum_data_x);
title('水平像素累加','FontSize',15);
index_x_1_1 = 0;
index_x_1_2 = 0;
index_x_2_1 = 0;
index_x_2_2 = 0;
l_temp = 20;
% Difference Method
Delta_sum_data_x = sum_data_x(1:end-l_temp)-sum_data_x((l_temp+1):end);
% Determining the Vertical Range From the Peak and Valley Values
figure;
plot([1:1:fc4_m-l_temp],Delta_sum_data_x);
title('水平像素累加-差分','FontSize',15);
```

```
% Choose

[pks,locs] = findpeaks(Delta_sum_data_x);

[ans_pks,pos] = sort(pks);

index_x(1) = locs(pos(end));

index_x(2) = locs(pos(end-1));

[pks,locs] = findpeaks(-Delta_sum_data_x);

[ans_pks,pos] = sort(pks);

index_x(3) = locs(pos(end));

index_x(4) = locs(pos(end-1));

index_x = sort(index_x,'ascend');

index_x_1_1 = index_x(1);

index_x_1_2 = index_x(2)+10;

index_x_2_1 = index_x(3);

index_x_2_2 = index_x(4)+10;
```

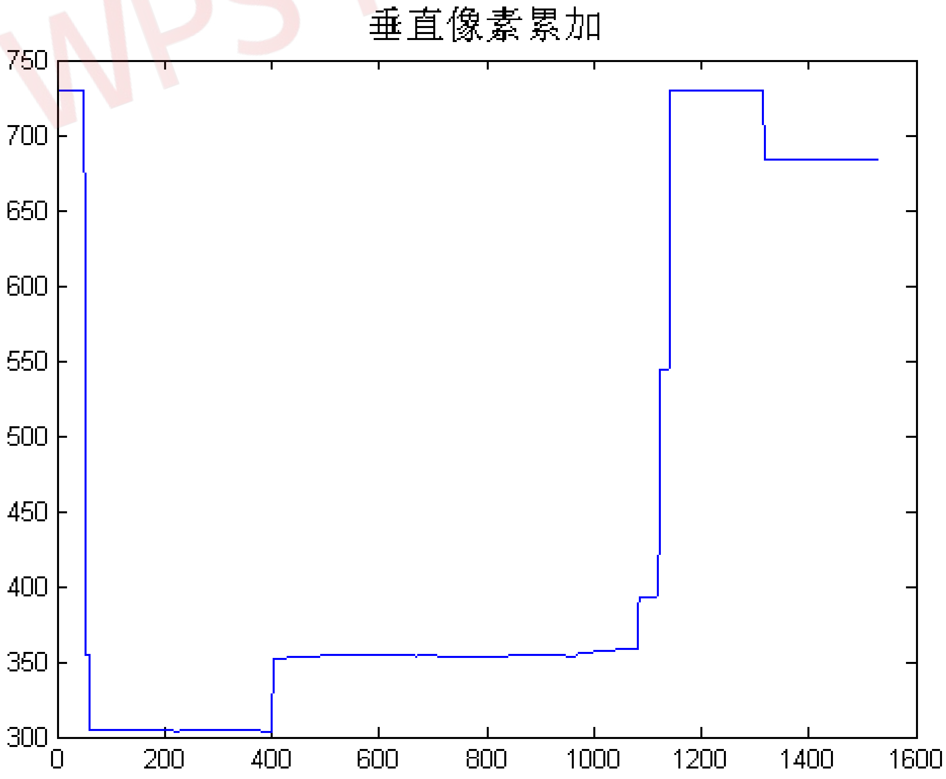


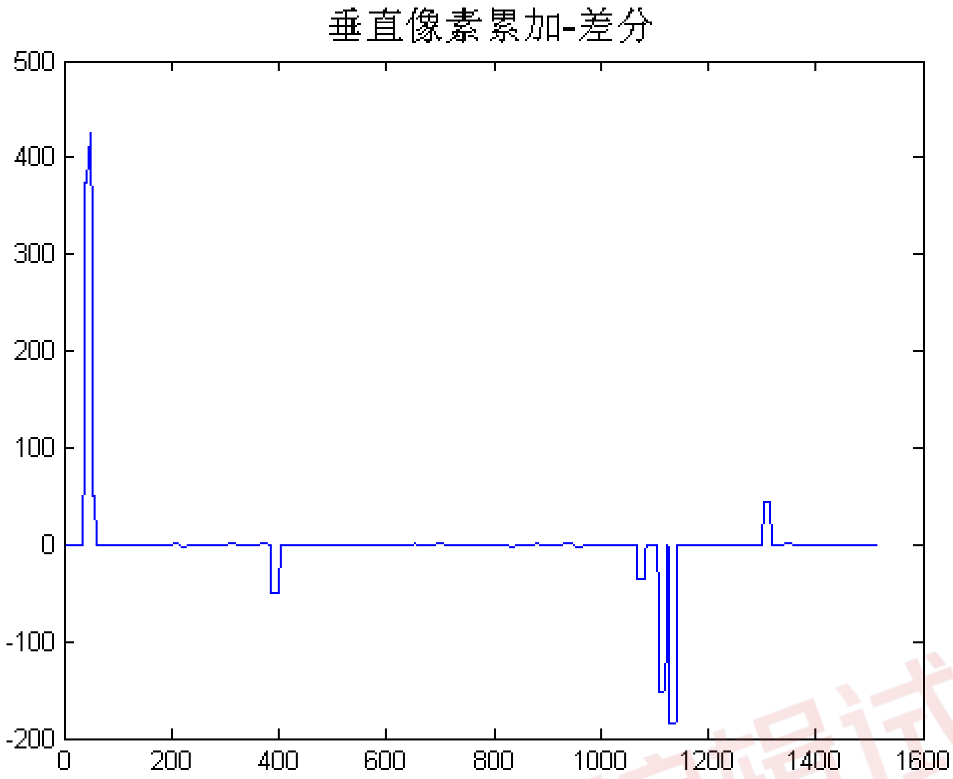


Horizontal Range

```
for i=1:1:fc4_n

sum_data_y(i) = sum(fc4(:,i));

end

figure;

plot([1:1:fc4_n],sum_data_y);

title('垂直像素累加','FontSize',15);

index_y_1 = 0;

index_y_2 = 0;

l_temp = 15;

% Difference Method

Delta_sum_data_y = sum_data_y(1:end-l_temp)-sum_data_y((l_temp+1):end);

% Determining the Horizontal Range From the Peak and Valley Values

figure;
```

```
plot([1:1:fc4_n-l_temp],Delta_sum_data_y);
title('垂直像素累加-差分','FontSize',15);
[pks,locs] = findpeaks(Delta_sum_data_y);
if max(pks) < 20
    index_y(1) = 1;
else
    [ans_pks,pos] = sort(pks,'ascend');
    index_y(1) = locs(pos(end));
end
[pks,locs] = findpeaks(-Delta_sum_data_y);
[ans_pks,pos] = sort(pks,'ascend');
index_y(2) = locs(pos(end))* 1.03;
index_y = sort(index_y,'ascend');
index_y_1 = index_y(1);
index_y_2 = index_y(2);
```

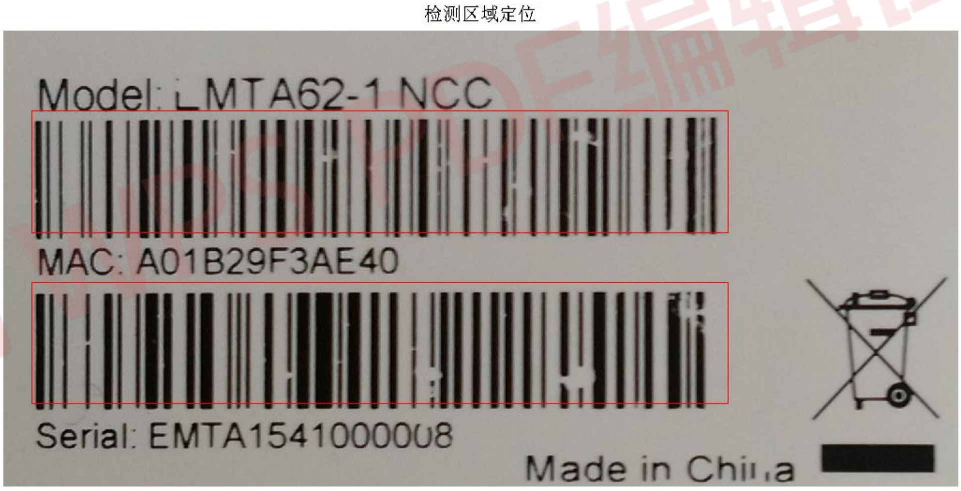




Positioning the Detection Area

```
A_a = [index_y_1,index_x_1_1];

A_b = [index_y_2,index_x_1_1];

A_c = [index_y_2,index_x_1_2];

A_d = [index_y_1,index_x_1_2];



B_a = [index_y_1,index_x_2_1];

B_b = [index_y_2,index_x_2_1];

B_c = [index_y_2,index_x_2_2];

B_d = [index_y_1,index_x_2_2];



A_x=[A_a(1),A_b(1),A_c(1),A_d(1)];

A_y=[A_a(2),A_b(2),A_c(2),A_d(2)];
```

```
v = [A_b; A_c; A_d ; A_a];



figure;imshow(im);title('检测区域定位','FontSize',15);hold on;

plot([A_a(1) A_b(1)],[A_a(2) A_b(2)],'r-');

plot([A_c(1) A_b(1)],[A_c(2) A_b(2)],'r-');

plot([A_c(1) A_d(1)],[A_c(2) A_d(2)],'r-');

plot([A_a(1) A_d(1)],[A_a(2) A_d(2)],'r-');



plot([B_a(1) B_b(1)],[B_a(2) B_b(2)],'r-');

plot([B_c(1) B_b(1)],[B_c(2) B_b(2)],'r-');

plot([B_c(1) B_d(1)],[B_c(2) B_d(2)],'r-');

plot([B_a(1) B_d(1)],[B_a(2) B_d(2)],'r-');
```



Crop Out the Region of Interest

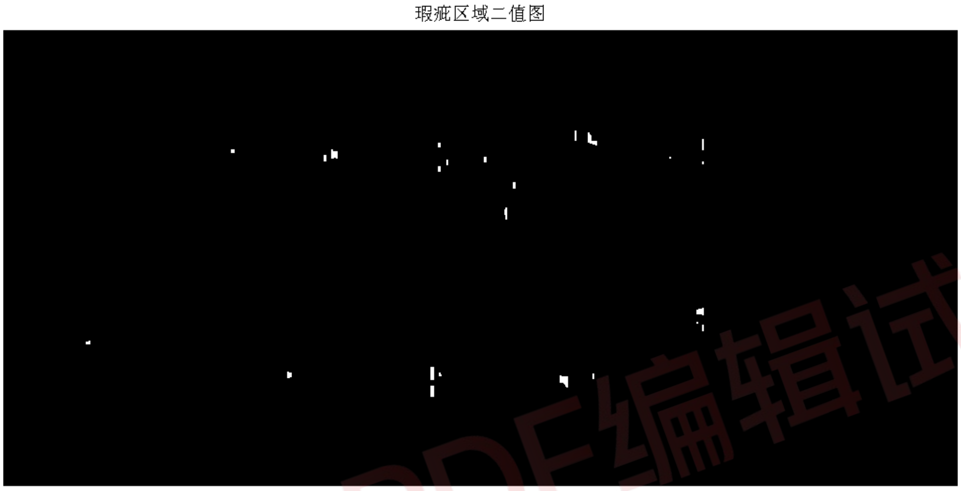
```
zero_one=zeros(fc4_m,fc4_n);

for i=1:fc4_m

  for j=1:fc4_n

     if (j>index_y_1)&&(j<index_y_2)&&(i>index_x_1_1)&&(i<index_x_1_2)
```

```
         zero_one(i,j) = 1;
      end
      if (j>index_y_1)&&(j<index_y_2)&&(i>index_x_2_1)&&(i<index_x_2_2)
         zero_one(i,j) = 1;
      end
   end
end
bw = bw .* zero_one;
for i=1:fc4_m
   for j=1:fc4_n
      if (j>index_y_1)&&(j<index_y_2)&&(i>index_x_1_1)&&(i<index_x_1_2)
      elseif (j>index_y_1)&&(j<index_y_2)&&(i>index_x_2_1)&&(i<index_x_2_2)
      else
         bw(i,j) = logical(1);
      end
   end
end
```

Defective areas

```
% Close
se = strel('rectangle',[6,2]);
fc_true = imclose(bw,se);


% Fill
sel =  strel('rectangle',[30,1]);
fc_fill  =imopen(fc_true,sel);
fc_fill  =imclose(fc_fill,sel);
```

```
e = fc_true-fc_fill;


se = strel('square',3);

fe = imopen(e,se);

figure;imshow(fe);title('瑕疵区域二值图','FontSize',15);
```

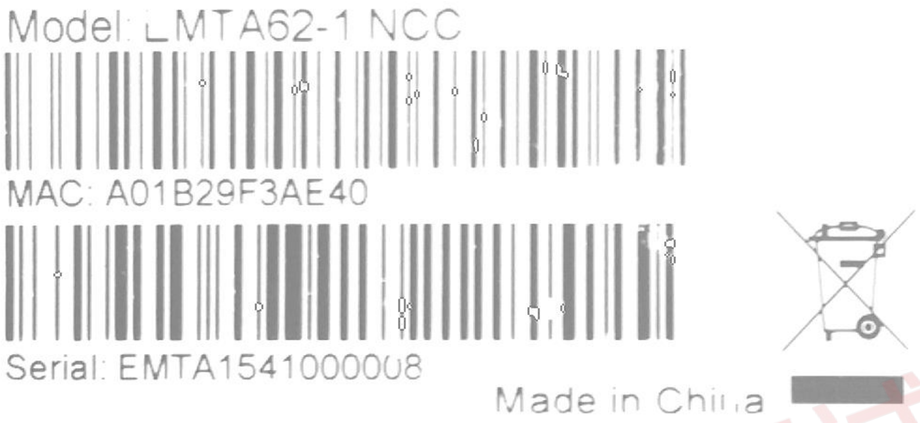


Locating Defective Areas

```
p=ones(fc4_m,fc4_n);

p = logical(p);

 for i=4:fc4_m-3

     for j=4:fc4_n-3

         if(fe(i,j)==0)&&(fe(i+3,j)==1||fe(i,j-3)==1||fe(i,j+3)==1||fe(i-3,j)==1)

           p(i,j) = 0;

         end

     end

 end

fc_temp = int8(double(im_gray).*double(p));
```

```
figure;

imshow(fc_temp);

title('定位瑕疵区域','FontSize',15)
```

定位瑕疵区域



Use the Red Frame to Circle the Defective Area

```
fe = logical(fe);

stats = regionprops(fe, 'BoundingBox' ,'Area','Centroid' ,'PixelList' );

centroids = cat(1, stats.Centroid);

noiseArea=1;

figure;imshow(im);

hold on

for i=1:size(stats)

    area = stats(i).Area;

    if (area > noiseArea) && ( stats(i).BoundingBox(3) > 2)

        x=centroids(i,1);y=centroids(i,2);

        % Width

        Width = stats(i).BoundingBox(3)* 5;

        % Length
        Length = stats(i).BoundingBox(4)* 5;

        rectangle  ('Position',[x-Width/2  y-Length/2  Width  Length],  'LineWidth',  2,

'LineStyle','-','EdgeColor','r');

        % pause(0.5)

        plot(centroids(i,1), centroids(i,2), 'b*');

    end

    hold on;

end

title('瑕疵检测结果','FontSize',15)
```

瑕疵检测结果