码字提取

PDF417的结构

0 -> 黑色
1 -> 白色
-> 模块
-> 条(1-6个黑色模块)
-> 空(1-6个白色模块)

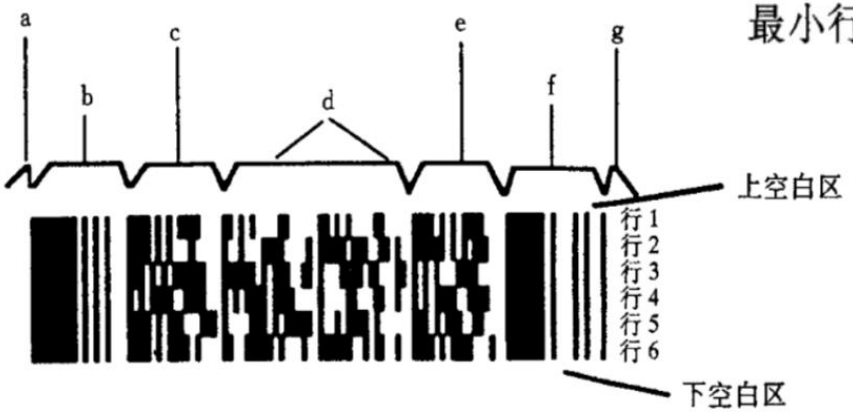
- 每个符号由4个条和4个空构成(从左到右:条->空->条…)
- 每个符号中总共有17个模块
- 起始符:8-1-1-1-1-1-1-3, 终止符:7-1-1-3-1-1-1-2-1

四一七条码使用簇号 0,3,6。簇号的定义适用于所有四一七条码符号字符。

四一七条码符号的每行只使用一个簇中的符号字符。同一簇每三行重复一次。第一行使用第 0 簇的符号字符,第 2 行使用第 3 簇的符号字符,第三行使用第 6 簇的符号字符,第四行使用第 0 簇的符号字符,以此类推。行号由上向下递增,最上一行行号为 1。

在每一簇中,每一符号字符对应唯一的码字,其范围为 0～928

a) 左空白区
b) 起始符
c) 左行指示符号字符
d) 1～30 个数据符号字符
e) 右行指示符号字符
f) 终止符
g) 右空白区



最小行数为 3,最大行数为 90

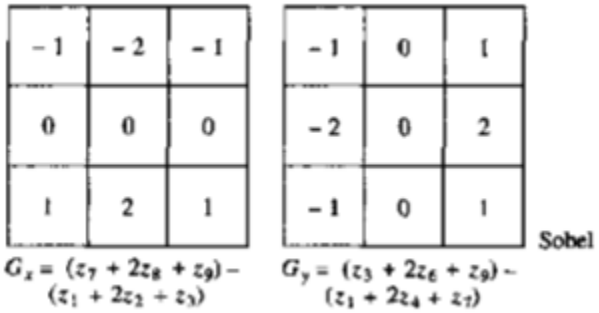
- 行指示符号字符 -
行号
行数
数据列数
纠错等级

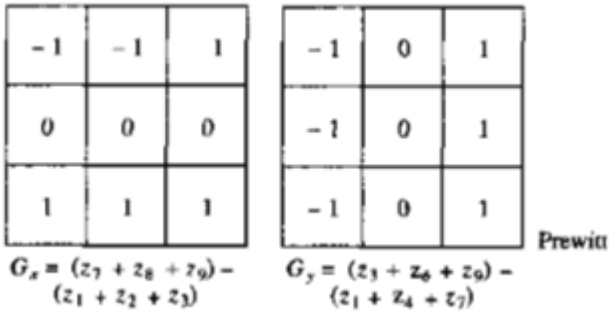
判断PDF417的层数

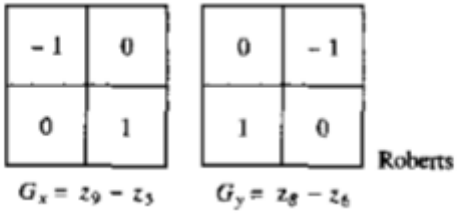
提取PDF417码字的第一步是先判断417码的层数。

判断方法如下：

对截取后的417码进行边缘提取。边缘提取算子可以用Sobel，Prewitt，Roberts等。







图像边缘:

- 利用计算机进行图像处理有两个目的：一是产生更适合人观察和识别的图像；二是希望能由计算机自动识别和理解图像。无论为了哪种目的，图像处理中关键的一步就是对包含有大量各式各样景物信息的图像进行分解。分解的最终结果是图像被分解成一些具有某种特征的最小成分，成为图像的基元。相对于整幅图像来说，这种基元更容易被快速处理。
- 图像的边缘是图像的最基本特征。所谓边缘(或边沿)是指其周围像素灰度有阶跃变化或屋顶变化的那些像素的集合。边缘广泛存在于物体与背景之间、物体与物体之间、基元与基元之间。因此，它是图像分割所依赖的重要特征。

图像边缘

- 经典的边缘提取方法是考察图像的每个像素在某个邻域内灰度的变化，利用边缘临近一阶或二阶方向导数变化规律，用简单的方法检测边缘。这种方法称为边缘检测局部算子法。
- 但在计算机处理过程中难以处理连续量导数，因此采用了离散的近似。如:

  $F'[n] = F[n] - F[n-1]$
- 由此产生出了最基本的图像边缘检测算子：Sobel算子:

-1	-2	-1
0	0	0
1	2	1

-1	0	1
-2	0	2
-1	0	1

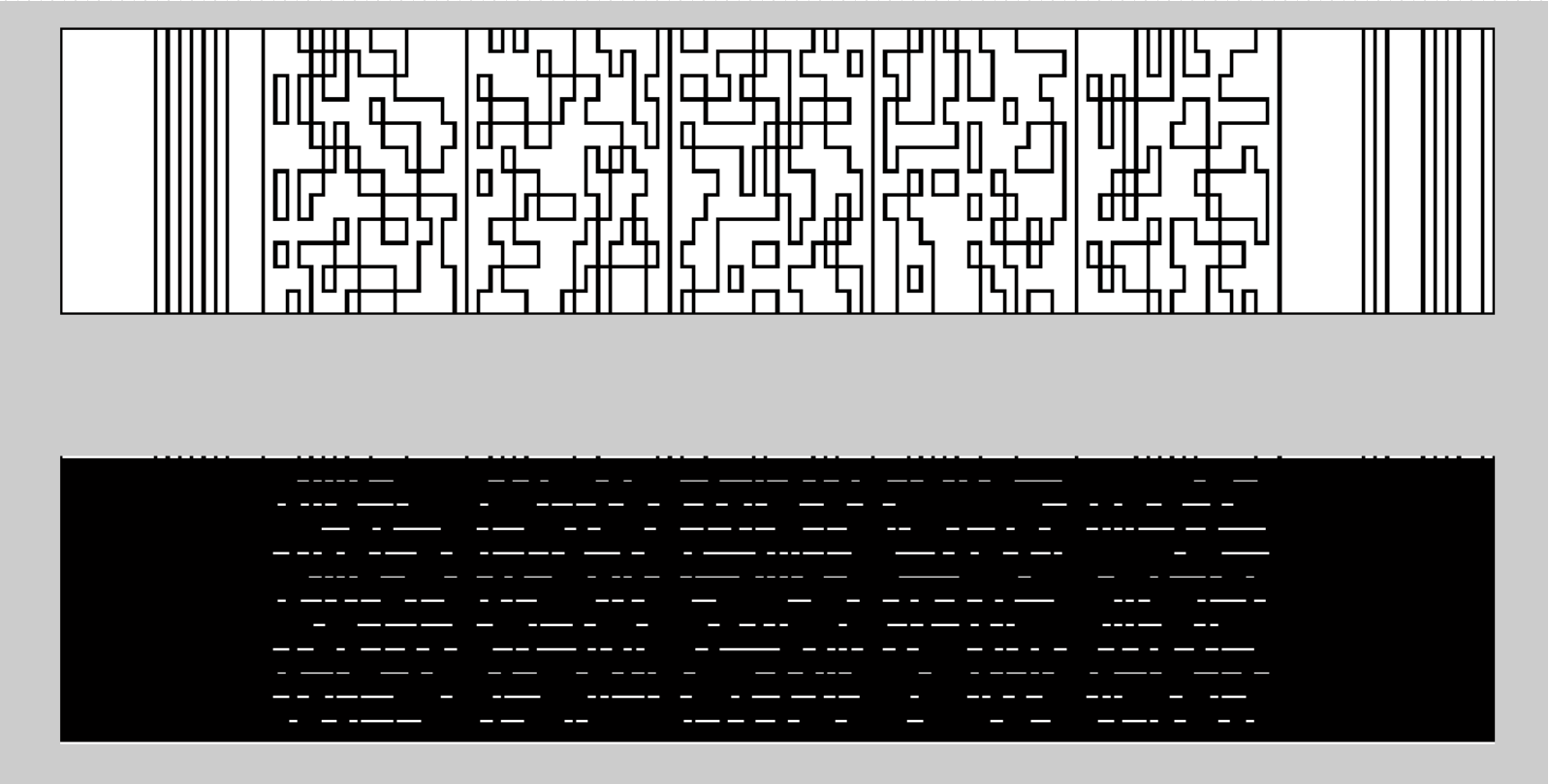
图像边缘

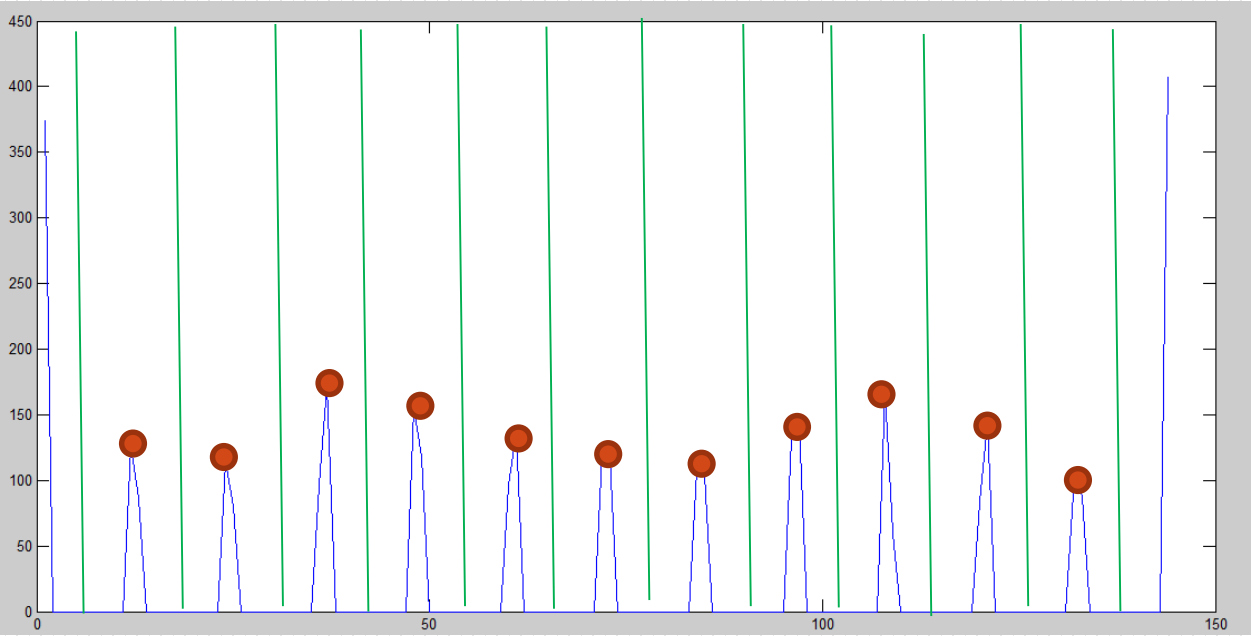
- Sobel边缘检测

图像中的每个点都用这两个核做卷积，一个核对通常的垂直边缘相应最大，而另一个对水平边缘相应最大。两个卷积的最大值作为该点的输出位。

```
%% 边缘检测算子
topso=[1 1 1;0 0 0;-1 -1 -1];
%botso=[-1 -1 -1;0 0 0;1 1 1];
lefso=[1 0 -1;1 0 -1;1 0 -1];
%rigso=[-1 0 1;-1 0 1;-1 0 1];
for xxx  %遍历图像每一点(!)
      %取出以该点为中心，与算子同样大小的区域temp
      %垂直方向 topso与temp点乘再将矩阵每个值加起来，记为tbs
      %水平方向 lefso与temp点乘再将矩阵每个值加起来，记为lbs
      if (tbs==0) && (lrs==0) %思考：这种情况是什么意思?
         %将该点赋值为255;
   end
end
```

- 提取到图像的边缘后，消除边缘图像的垂直边缘。然后做水平投影。





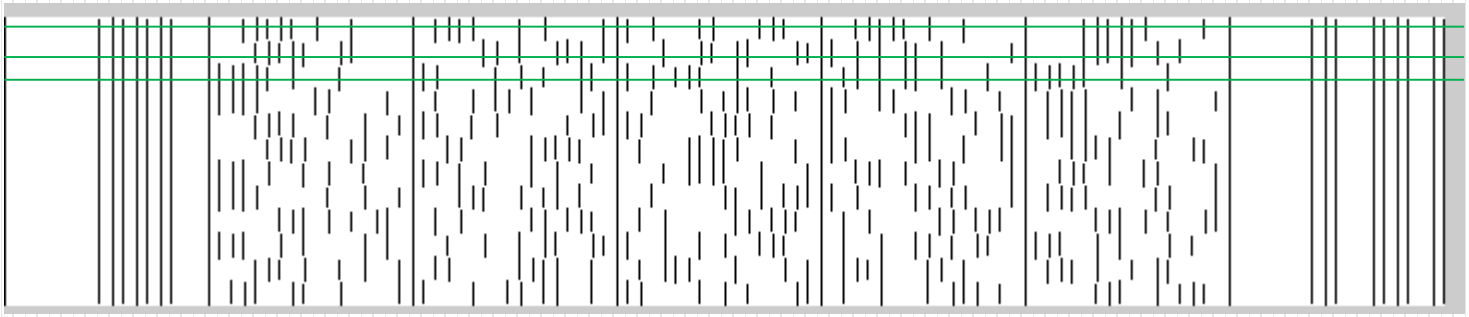
可以看到，图像中层与层之间的边界都会在投影图上形成明显的峰值。统计这些峰值（图上红点标注的点）所在的位置到一个数组line[]上。

根据line数组，计算每一层的中心（绿线标注的位置）所在的行数到数组Layers[]上

```
%% 判断数据码字行数
for xxx  %遍历每一行(!)
   if %当前行比它的上n行和下m行的值都大，则判断为峰值
      k=k+1;
      line(k)=i;
   end
end
k %层数k
```

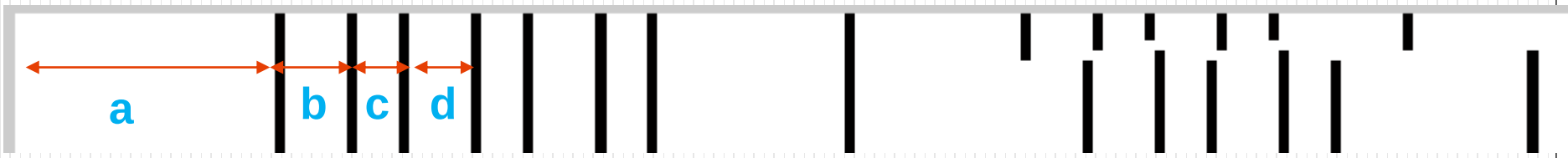
提取码字

消除边缘图像中的水平边缘



建立一个矩阵code[L][col] L是层数。

根据layers[]来在绿线上遍历图像。



当src(layer(i),j)==0　i<L, 2<j<col，即遍历到垂直的边缘时记录此像素和上一个记录点的距离。

上图以第i排为例code(i,1:4)={a,b,c,d}。

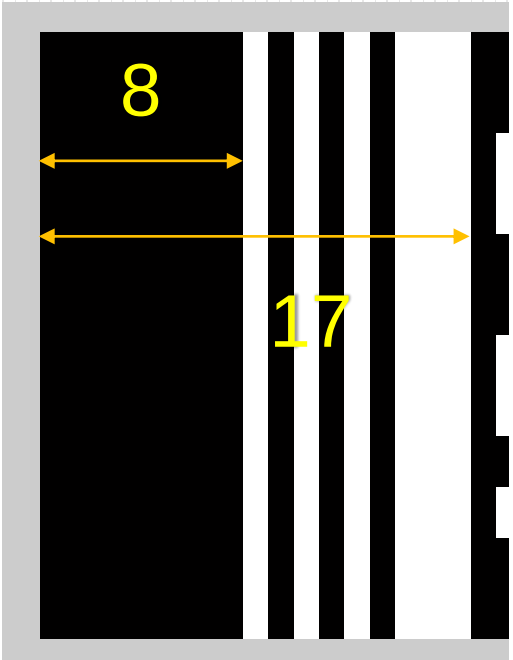
遍历后去除code[][]中的零向量。(为什么？)

	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69
1	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
2	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
3	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
4	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
5	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
6	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
7	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
8	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
9	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
10	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
11	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
12	5	7	5	13	5	0	0	0	0	0	0	0	0	0	0	0	0
13																	
14																	
15																	
16																	
17																	

```
minst=zeros(1,L);
codes=zeros(L,col);
for i=1:L
    m=0;
    for j=2:col
        if %当前点为0
            if m==0
                m=m+1;
                %在codes中记录该点位置;
                temp=j;
            else
                m=m+1;
                %在codes中记录该点与上一个点的距离;
                temp=j;
            end
            % 在minst中记录改行的点数;
        end
    end
end
```

计算模块的大小

- 计算PDF417中一个码字由17个模块组成。起始符的长是17个模块，前面黑色那块长是8个模块。
- 计算矩阵code中第一列的平均值，除以8，四舍五入后即为一个模块的长度aunit。
- Code数组点除以aunit，四舍五入后就能等得到期望的符号码字。



解码码字转换

16

矩阵code中，前17列和后17列分别是起始符和终止符，这两个符号不携带码字，可以除去。

Row，Col 分别是去除起始和终止符号后code的行数与列数。遍历矩阵code，每从(j,i)处取8个数

$$\text{temp} = \sum_{k=j}^{j+7} code(i,k) * 10\text{^}(8-k)$$

历遍PDF417的符号转换表，在其中找出temp所对应的码字，用一个数组decode来记录。

用load 命令来读取.mat类型的数据

```
1 -    load symcodes.mat -ascii
```

GB/T 17172—1997

表 A1　符号字符—码字集

第0簇											
bsbsbsbs	码字	bsbsbsbs	码字	bsbsbsbs	码字	bsbsbsbs	码字	bsbsbsbs	码字	bsbsbsbs	码字
31111136	0	41111144	1	51111152	2	31111235	3	41111243	4	51111251	5
21111326	6	31111334	7	21111425	8	11111516	9	21111524	10	11111615	11
21112136	12	31112144	13	41112152	14	21112235	15	31112243	16	41112251	17
11112326	18	21112334	19	11112425	20	11113136	21	21113144	22	31113152	23
11113235	24	21113243	25	31113251	26	11113334	27	21113342	28	11114144	29
21114152	30	11114243	31	21114251	32	11115152	33	51116111	34	31121135	35
41121143	36	51121151	37	21121226	38	31121234	39	41121242	40	21121325	41
31121333	42	11121416	43	21121424	44	31121432	45	11121515	46	21121523	47
11121614	48	21122135	49	31122143	50	41122151	51	11122226	52	21122234	53
31122242	54	11122325	55	21122333	56	31122341	57	11122424	58	21122432	59
11123135	60	21123143	61	31123151	62	11123234	63	21123242	64	11123333	65
21123341	66	11124143	67	21124151	68	11124242	69	11124341	70	21131126	71
31131134	72	41131142	73	21131225	74	31131233	75	41131241	76	11131316	77

解码结果

	n值		
m=1	32	902	3
m=2	393	512	563
m=3	733	323	226
m=1	640	819	496
m=2	645	900	883
m=3	821	459	841
m=1	278	151	62
m=2	131	885	899
m=3	899	899	899
m=1	883	899	853
m=2	901	255	41
m=3	777	735	806

```
for %通过设置步长遍历图中每个码字
    temp = acodes(i,j)*10000000 + acodes(i,j+1)*1000000.....%转换为数字码字
    for xxx %遍历symcodes寻找对应码字
        if %temp与symcodes其中的码字一致
            num=num+1;
            dcodes(num)=n-1;
            break;
        end
    end
```

迷人风景只有站在一定高度才能领略，望诸君努力，谢谢！