数字图像处理实验三：

- 学号：161220129
- 姓名：王奕琛
- 邮箱：919345923@qq.com
- 时间：2019/5/28

实验思路：

- 识别数字：
- 首先从example.png图像中截取0-9是个数字的图像并存储为0-9.jpg，作为比较用的模板。
  - 读入待识别的图像，将待识别图像与模板图像转化为二值图像方便后续处理。
- 将待识别图像与模板图像进行去边话处理，即寻找其有效图像所对应的最小外接矩形。
  - 将模板图像进行缩放使之与待识别图像大小一致。
  - 逐一比较模板图像与待识别图像的每个像素点，得到个数比$\frac{\text{相同像素点}}{\text{总像素点}}$,则使该数值较大的模板图像所对应的数字即为待识别图像的数字
- 识别符号：
  - 基本与识别数字相同
  - 在取最小外接矩形后，由于加号，减号，等号所对应的图像长宽比有较大的区别，此处可以进行优化，通过对长宽比的值的判断来判断符号类
- 计算图像中的加减法
  - 将图像进行二值化处理
  - 将图像按照黑色线条切割为30个小的矩形图像分别出了(计算每一行的像素值的和，小于特定值即为黑色线条所在行)
  - 对于每个小的矩形图像，再按照特定值切割得到两个只含一个数字的图像与一个只含一个符号的图像(此处切割直接指定值)。
  - 再对每个切割得到的图像进行识别，并按照运算规则进行运算，得到结果。
  - 将结果对应的模板图像缩放至一定大小后加入小的矩形图像上等号后的位置(直接指定位置)。

代码实现：

- 识别数字

```
function output = my_digit(input_image)

%对输入图像进行去边化处理
input_image=minboundary(input_image);
%计算与每一个模板图像的相似度
jud=judge(input_image);
max=0;pos=0;
for index=1:10
    if(max < jud(index))
```

```
        max=jud(index);
        pos=index;
    end
end
%取相似度最大值对应的下标-1作为返回值
output = int2str(pos-1);

    %去边化处理
    function output = minboundary(input)
        input2 = input;
        k=0;
        Size = size(input);
        rows = Size(1);
        cols = Size(2);
        for i=1:rows
            %改边是否含有效图像
            flag=0;
            for j=1:cols
                %二值图像，含有效图形则像素值为0
                if(input(i,j)==0)
                    flag=1;
                end
            end
            if(flag==0)
                input2(i-k,:)=[];
                k=k+1;
            end
        end
        k=0;
        for i=1:cols
            flag=0;
            for j=1:rows
                if(input(j,i)== 0)
                    flag=1;
                end
            end
            if(flag==0)
                input2(:,i-k)=[];
                k=k+1;
            end
        end
        output=input2;
    end
    %计算与每个模板图像的相似度
    function output = judge(input)
        judge=zeros(1,10);
        %读入模板图像
        img0=imread('0.jpg');
        img0=rgb2gray(img0);    img0=imbinarize(img0);
```

```
        %计算相似度
        judge(1,1) = similar(img0,input);

        %其他模板图像的读入与1相同，不再加入
        %...
        %...

        %相似值存入judge中
        output=judge;
    end

    function output = similar(input1,input2)
        t1=minboundary2(input1);
        %imshow(t1);
        t2=input2;
        [a1,b1]=size(t2);
        %缩放，使大小一致
        t1=imresize(t1,[a1,b1],'bicubic');
        t2=imresize(t2,[a1,b1],'bicubic');
        sum=0;
        for i=1:a1
            for j=1:b1
                if(t1(i,j)== t2(i,j))
                    %两像素点相同，则sum+1,
                    sum=sum+1;
                end
            end
        end
        %sum/总点数作为相似度
        output = sum/(a1*b1);
    end

    function output=minboundary2(bw)
        %对模板图像的去边化处理，与minboundary基本相同，在调试过程中添加且最终未删除。事实上可以与minboundary合并。
    end

end
```

- 识别符号(与识别数字相同)
- 运算

```
function output = my_calculator(input_image)

%先转化为二值图像
bw = imbinarize(input_image);
imgSize = size(bw);

imgWidth = imgSize(1);
```

```
rows = ones(1,imgWidth);
imgHeight = imgSize(2);
cols = ones(1,imgHeight);

%获取各行列的像素值之和
for i=1:imgWidth
    for j=1:imgHeight
        x=rows(1,i);
        y=bw(i,j);
        A=uint32(x)+uint32(y);
        rows(1,i)=A;
        uint32 B;
        B=uint32(cols(1,j))+uint32(bw(i,j));
        cols(1,j)=B;
    end
end


%像素值之和小于一定数值表示该行或列为黑色直线，将其标注出存入R,C中
R = zeros(11,1);
C = zeros(4,1);
k = 1;
for i=1:imgWidth
    if((rows(i) < 100) && (k==1||((i-R(k-1))>10)))
        R(k) = i;
        k=k+1;
    end
end
k=1;
for i=1:imgHeight
    if(cols(i) < 100 && (k==1 || (i-C(k-1)>10)))
        C(k) = i;
        k=k+1;
    end
end

%此时以R,C中的值为分割位置，对原图进行分割。
for i=1:10
    for j=1:3
        %按照原图的线条得到30个小的图像，并分别对这30个小图进行处理。
        image = zeros(R(i+1)-R(i)+1,C(j+1)-C(j)+1);
        for ii=R(i):R(i+1)
            for jj=C(j):C(j+1)
                image(ii-R(i)+1,jj-C(j)+1) = bw(ii,jj);
            end
        end

        %对每个小图像进行运算处理
        newImg=franImage(image);
```

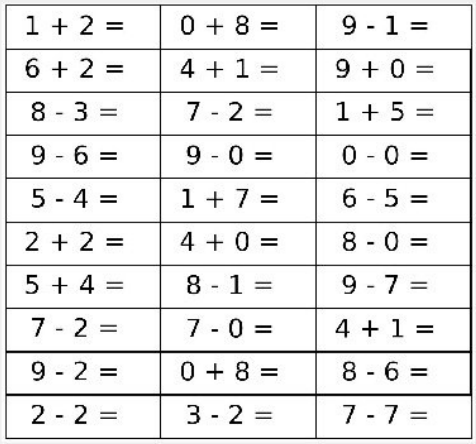
```
        %用运算后的图像替代初始图像
        for ii=R(i):R(i+1)
            for jj=C(j):C(j+1)
                bw(ii,jj) = newImg(ii-R(i)+1,jj-C(j)+1);
            end
        end
    end
end

%二值图像恢复为原图像
for i=1:imgWidth
    for j=1:imgHeight
        if(bw(i,j)==1)
            outImg(i,j)=255;
        end
    end
end
output=outImg;

    %处理分割的小矩形图像
    function output = franImage(f_image)

        %获取第一个数字
        dig1 = ones(70,60);
        for i1=11:80
            for j1=31:90
              dig1(i1-10,j1-30) = f_image(i1,j1);
            end
        end
        d1 = my_digit(dig1);

        %获取操作符
        oper = ones(70,60);
        for i1=11:80
            for j1=91:140
                oper(i1-10,j1-90) = f_image(i1,j1);
            end
        end
        op1 = my_operator(oper);

        %获取第二个数字
        dig2 = ones(70,60);
        for i1=11:80
            for j1=151:200
                dig2(i1-10,j1-150) = f_image(i1,j1);
            end
        end
        d2 = my_digit(dig2);
```

```
        %将数字转为int
        x1=str2int(d1); y1=str2int(d2);
        result=0;
        %判断符号，进行运算
        switch op1
            case'+'
                result=x1+y1;
            case'-'
                result=x1-y1;
            otherwise
                result=0;
        end

        %得到结果对应的图像
        numImg=getimage(result);

        %将结果贴到原图像上
        for i1=11:80
            for j1=251:310
                f_image(i1,j1)=numImg(i1-10,j1-250);
            end
        end
        output = f_image;
    end

    %字符转数字
    function output = str2int(input)
        switch input
            case '1'
                output=1;
            %...
            %...
            otherwise
                output=0;
        end
    end

    %得到数字对应的图像
    function output = getimage(input)
        switch input
            case 1
                imgNum=imread('1.jpg');
            %...
            %...
            otherwise
                imgNum=imread('0.jpg');
        end
        imgNum=rgb2gray(imgNum);
```

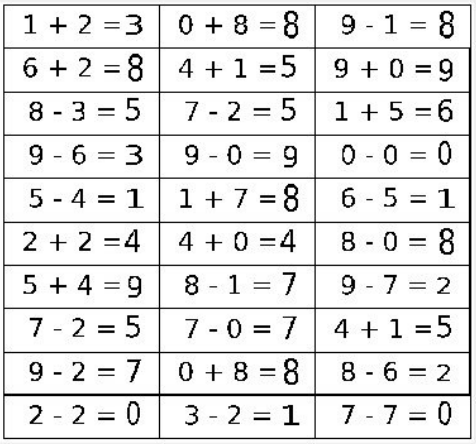
```
        imgNum=imbinarize(imgNum);
        output=imresize(imgNum,[70,60],'bicubic');
    end

end
```

处理结果：

- 由于每个运算结果的图形是手动裁剪的，结果美观度不足(对example.png的处理结果)



1 + 2 =	0 + 8 =	9 - 1 =
6 + 2 =	4 + 1 =	9 + 0 =
8 - 3 =	7 - 2 =	1 + 5 =
9 - 6 =	9 - 0 =	0 - 0 =
5 - 4 =	1 + 7 =	6 - 5 =
2 + 2 =	4 + 0 =	8 - 0 =
5 + 4 =	8 - 1 =	9 - 7 =
7 - 2 =	7 - 0 =	4 + 1 =
9 - 2 =	0 + 8 =	8 - 6 =
2 - 2 =	3 - 2 =	7 - 7 =



1 + 2 =3	0 + 8 =8	9 - 1 = 8
6 + 2 =8	4 + 1 =5	9 + 0 =9
8 - 3 = 5	7 - 2 = 5	1 + 5 =6
9 - 6 = 3	9 - 0 = 9	0 - 0 = 0
5 - 4 = 1	1 + 7 =8	6 - 5 = 1
2 + 2 =4	4 + 0 =4	8 - 0 = 8
5 + 4 =9	8 - 1 = 7	9 - 7 = 2
7 - 2 = 5	7 - 0 = 7	4 + 1 =5
9 - 2 = 7	0 + 8 =8	8 - 6 = 2
2 - 2 = 0	3 - 2 = 1	7 - 7 = 0