数字图像处理 作业三

字符识别：从图片中计算加减算式结果

一、实验完成情况

完成了全部题目要求，程序可以单独识别数字图片和运算符图片，也可以处理输入的包含加法和减法算式的图片，并输出包含计算结果的图片（输出到屏幕的同时，保存到 **/asset/image** 文件夹中）。

实验平台：Windows 10 + MATLAB R2016b

测试方法（my_test.m）

- 测试 **my_digit** 函数（数字识别）

```
imgInput = imread('digit7.jpg');
ib = imbinarize(rgb2gray(imgInput));
digit = my_digit(ib);
figure
imshow(ib);
title(['识别结果: ', digit],'FontSize',20);
```

- 测试 **my_operator** 函数（运算符识别）

```
imgInput = imread('op+.jpg');
ib = imbinarize(rgb2gray(imgInput));
op = my_operator(ib);
figure
imshow(ib);
title(['识别结果: ', op],'FontSize',20);
```

- 测试 **my_calculator** 函数（算式求解）

```
imname = 'test1.png';
imgInput = imread(imname);
imgOutput = my_calculator(imgInput);
imname = ['E:/Workspace/DIP_Lab/Project3/asset/image/', imname(1:5), '_result.png'];
imwrite(imgOutput, imname);

subplot(1, 2, 1);
imshow(imgInput);
subplot(1, 2, 2);
imshow(imgOutput);
```

（在 **Windows** 系统中，无法使用 **../asset/image/** 的方式指定文件的保存路径，故将绝对路径写在代码中）

二、函数功能

下面，分别介绍程序中每个函数的功能。

- minboundrect.m

minboundrect 函数是早期版本的 MATLAB 中自带的库函数，后来它被 bwlabel 和 regionprops 所替代。但是，在某些场景下，直接调用 minboundrect 函数要更加简便。所以，我在本次实验中通过直接添加其代码[1]到本地的方式对其进行使用。

minboundrect 函数的功能是，对于输入的若干图片坐标点，输出一个包含这些坐标的最小矩形（"最小"可以由面积或者周长来度量）。

- cut.m

在本程序中，cut 函数用来除去传入的待计算图片周围的白边，以及将待计算图片划分为若干小块，每一块中包含一个算式，以方便识别和计算。

利用 minboundrect 函数，cut 函数可以简单地进行实现。具体请见代码。

- my_operator.m

本函数用于识别输入的二值图像中包含的运算符。实现思路如下：

首先调用 minboundrect 函数，找出包含运算符的矩形区域。由于三种运算符"+"，"-"与"="的形态差别非常明显，所以可以通过简单的条件判断来确定图片中是哪一种运算符。具体而言，如果是"="，那么矩形区域中间应当是白色像素；否则，说明图片中是"-"或者"+"。很容易观察到，包含"-"的最小矩形区域中的绝大部分应该都是黑色像素。通过这一点，可以进一步识别出"+"和"-"。
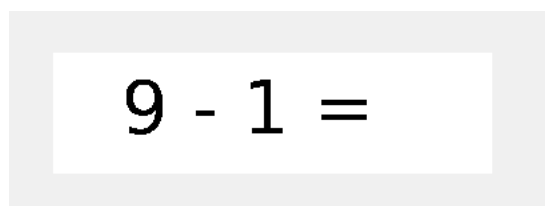
- my_digit.m

本函数用于识别输入的二值图像中包含的数字。

在开始实验的时候，我打算利用与 my_operator 类似的方法，利用数字的对称性和笔画的方向等特征，对 0-9 这十个数字进行检测。但是，我调试了很久，也没有找到方法，能够检测任意大小的、包含数字的图片，同时还要保证正确性。后来为了完成实验，我直接调用了 MATLAB 的 ocr 库[2]。

- my_calculator.m

本函数用于对输入的待计算图片进行计算并输出结果。

首先，调用 cut 函数截去图像的白边。我们的测试图片中共有 10 行 3 列，分为 30 个相同的格子，每个格子中有一个待计算的式子。于是，我们可以求出每个格子的高和宽，进而确定每个格子（矩形）四个顶点的位置。调用 cut 函数即可把每个格子切分出来。如下所

示:

<p align="center">切分出的一个 grid</p>

接着，使用 MATLAB 的库函数 bwlabel 和 regionprops，可以确定 grid 中各个连通域的位置。每个 grid 中应当有 5 个连通域（等号为两个连通域）。将前三个连通域所在的矩阵提取出来，作为包含数字和运算符的图像，传入 my_digit 和 my_operator 函数中。获得两个运算数和运算符分别是什么之后，直接计算就可以了。

根据要求，计算出的结果还需要显示在图片上。这里，我的处理方式是，预先从测试图片中截取 80x80 像素的各个数字的图片，保存为 digit0~digit9.jpg。当运算结果为 k 时，直接读取 digitk.jpg，转化为二值图像，复制到原图像的相应位置即可。

my_calculator 函数的实现思路是比较简单的，没有用到复杂的技术，具体的实现请参见代码。

三、实验结果

my_digit 函数和 my_operator 函数均能够正确识别数字图片和运算符图片，并返回相应的字符类型变量，
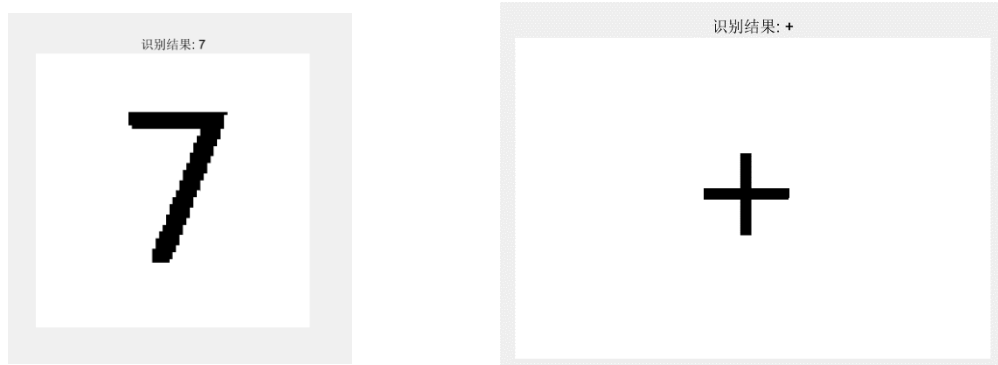
对题目所给的六张待计算图片(test1~6)进行了测试，均能得到正确的结果，如下所示：

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

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

test1

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

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

test2

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

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

test3

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

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

test4

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

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

test5

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

test6

参考资料

1. https://www.mathworks.com/matlabcentral/mlc-downloads/downloads/submissions/34767/versions/3/previews/MinBoundSuite/minboundrect.m/index.html

2. https://www.mathworks.com/help/vision/examples/recognize-text-using-optical-character-recognition-ocr.html

参考资料