图像处理报告一

1.1 基础的图像处理

1.1.1 函数列表

- **C = cat(dim,A1,A2,…,An) concatenates A1, A2, … , An along dimension dim.**

  You can use the square bracket operator [] to concatenate. For example, [A,B] or [A B] concatenates arrays A and B horizontally, and [A; B] concatenates them vertically.

1.1.2 代码及仿真结果

```
im = imread('apples.jpg');
imr = im(:,:,1); % get red channel
img = im(:,:,2); % get green channel
imb = im(:,:,3); % get blue channel
% crop 50 x 50 pixel region starting at (100,120)
im_crop = im(100:150,120:170,:);
figure
imshow(im_crop);
```

截取部分像素点

```
% swap the red and blue image channels
im2 = cat(3,im(:,:,3),im(:,:,2),im(:,:,1));
figure
imshow(im2);
```

交换 RGB 通道的颜色，等效于反色

```
% sub-sample every 2nd pixel in vertical direction and 5rd pixel in horizontal direction
im_subsamp = im(1:2:end,1:5:end,:);
figure
imshow(im_subsamp);
```

等间隔抽样，等效于压缩

```
im_flipped = im(end:-1:1,:,:); % flip image (vertical)
figure
imshow(im_flipped);
```

对第一维度（行）进行翻转，等效于垂直翻转

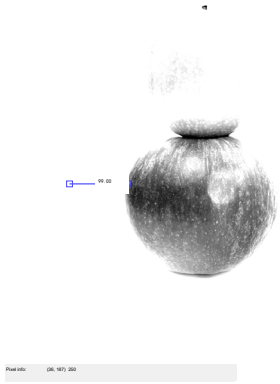
1.2 图像展示

1.2.1 函数列表

- **imshow(f, [low high])**
  all values in the image which are less than or equal to low are displayed as black, and all values greater than or equal to high are displayed as white. Values in between are displayed as intermediate intensity values using the default number of levels.
- **impixelinfo**
  create a Pixel Information tool,(x coordinate,y coordinate) pixel value
- **imdistline(gca,[a b],[c d])**
  sets the first endpoint at the (x,y) coordinate (a, c) and the second endpoint at the coordinate (b, d).
- **Impixelinfo**
  create a Pixel Information tool,(x coordinate,y coordinate) pixel value , always stands after function imshow()
- **h = imdistline(gca,[a b],[c d])**
  sets the first endpoint at the (x,y) coordinate (a, c) and the second endpoint at the coordinate (b, d)

1.2.2 代码及运行结果

```
clear all;close all;
f = imread('apples.jpg');
imshow(rgb2gray(f), [50 150])
% works even if all 3 channels are not identical, but then the original image is not greyscale
% imshow with a RGB image appears to ignore the 'DisplayRange' argument.
% imshow(f)
impixelinfo;
%create a Pixel Information tool,(x coordinate,y coordinate) pixel value
h = imdistline(gca,[1 100],[400 400])
% h = imdistline(gca,[a b],[c d])
% sets the first endpoint at the (x,y) coordinate (a, c) and the second endpoint at the coordinate (b, d).
% also can use imdistline directly;
```



1.3 写入图像及压缩

```
1. f = imread('apples.jpg');
2. imwrite(f,'C:\Users\yf\Documents\course\图像处理
   \week1\apples_copy.jpg','quality',20)
3. % K = imfinfo('apples_copy.jpg')
4. % imwrite(f,filename,'quality',q),q is an integer between 0 and 100 whicn in
   dicate compression degree
```

apples.jpg

apples_copy.jpg

1.4 图像数据类型及转换

1.4.1 函数列表

Function:	Convert input to:	Valid input data classes:
im2uint8	uint8	logical,uint8,uint16,double
im2uint16	uint16	logical,uint8,uint16,double
mat2gray	double (in range [0, 1])	double
im2double	double	logical,uint8,uint16,double
im2bw	logical	uint8,uint16,double

1.4.2 数据类型解析

matlab 中读取图片后保存的数据是 uint8 类型(8 位无符号整数，即 1 个字节)，以此方式存储的图像称作 8 位图像，好处相比较默认 matlab 数据类型双精度浮点 double（64 位，8 个字节），自然可以节省很大一部分存储空间。

详细来说 imread 把灰度图像存入一个 8 位矩阵，当为 RGB 图像时，就存入 8 位 RGB 矩阵中。例如，彩色图像像素大小是 400*300(高 * 宽)，则保存的数据矩阵为 400*300*3，其中每个颜色通道值是处于 0~255 之间。

但是虽然 matlab 中读入图像的数据类型是 uint8，而在图像矩阵运算的时候，使用的数据类型却是 double 类型。一是为了保证精度，二是因为如果不转换，在对 uint8 进行加减时会产生溢出。

matlab 读入图像的数据是 uint8，而 matlab 中数值一般采用 double 型（64 位）存储和运算。所以要先将图像转为 double 格式的才能运算

```
1.  I1  = im2double(img);    % 把图像转换成 double 精度类型（0~1）
2.  I2  = double(img)/255;   % uint8 转换成 double,作用同 im2double
```

补充说明一下，im2double( )和 double( )的区别。double(img)就是简单的数据类型转换，将无符号整型转换为双精度浮点型 double，但是数据大小没有变化，原本数据是 0~255 之间，转化后还是 0~255。例如原来是 255，那么转换后为 255.0，小数位 0 个数是由 double 数据长度决定，实际数据大小还是 255，只不过这个 255 已经是 double 类型空间存储了，再增加不会发生溢出情况。而 im2double(img)则不仅仅是将 uint8 转换到 double 类型，而且把数据大小从 0~255 映射到 0~1 区间。[1]

1.5 Image histograms

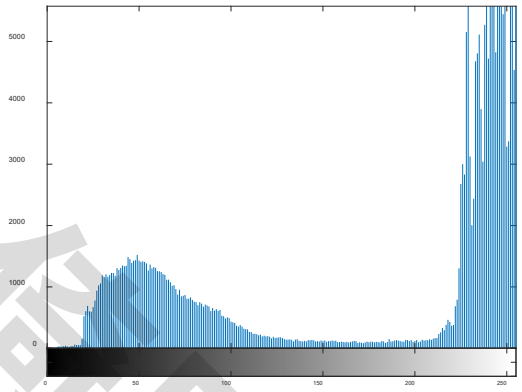
*1.*函数列表

- **Z = imlincomb(K1,A1,K2,A2,...,Kn,An)**
  computes the linear combination of images, A1, A2, … , An, with weights K1, K2, … , Kn accor ding to: Z = K1*A1 + K2*A2 + ... + Kn*An
- **imhist(im_gray)**
  plot a histogram for a single channel.

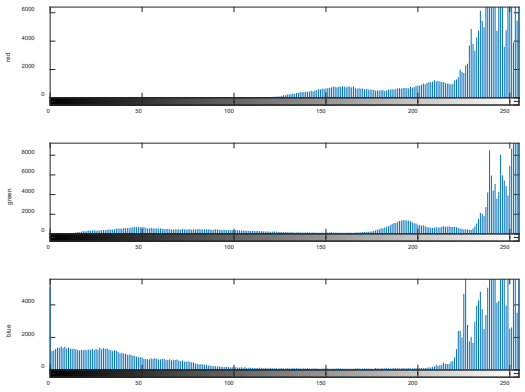
2.代码及运行结果

[1] 转载自 CSDN 博主「无鞋童鞋」的原创文章。
原文链接：https://blog.csdn.net/FX677588/article/details/53301740

```
clear all;close all;
im = imread('apples.jpg');
im_gray = imlincomb(0.1,im(:,:,1), 0.1,im(:,:,2), 0.8,im(:,:,3));
% Z = imlincomb(K1,A1,K2,A2,...,Kn,An)
% computes the linear combination of images, A1, A2, … , An, with weights K1 , K2, … , Kn according to:
% Z = K1*A1 + K2*A2 + ... + Kn*An
imhist(im_gray)
```



```
im = imread('apples.jpg');
figure;
subplot(3,1,1)
imhist(im(:,:,1)) % red channel
ylabel('red')
subplot(3,1,2)
imhist(im(:,:,2)) % green channel
ylabel('green')
subplot(3,1,3)
imhist(im(:,:,3)) % blue channel
ylabel('blue')
```



Tutorial Activity 1: Basic Image Operations

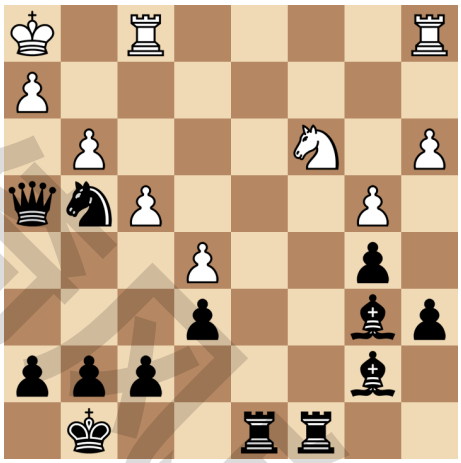
1. 思路分析：

题目要求：

a) 给棋盘分成均匀的 8×8 块区域，对应的行、列取值范围皆为 0 到 7 间的整数（包括端点值）

b) 判断某块棋盘内的棋子是黑棋、白棋，或是空白。

c) 画出对应棋盘的棋子



为了判断棋盘内是否有棋子，我们利用当棋盘内所有像素点的灰度值与方格左上角第一个像素点的灰度值相同时，该棋盘内没有棋子，否则有棋子。

当方格内有棋子时，为了判断其为黑棋抑或是白棋，我们以一个方格内白色像素点（灰度为 255）占总像素点的百分比为评判标准，称为 white_pixel，经过计算发现，当 white_pixel 大于 85%时，棋子判别的准确率达到 100%。

2. 代码与解析如下所示：

子函数 board_position(row,col)

```
1.  function im = board_position(row,col)
2.      % Load Image
3.      im = imread('chess.png');
4.      % Obtain the size of the image
5.      [rows, columns] = size(im);
6.
7.      % Calculate the board size
8.      board_size = rows/8;
9.      %divide the board into 64 area
10.
11.     % Calculate the require position
12.     row_start = row * board_size + 1;
```

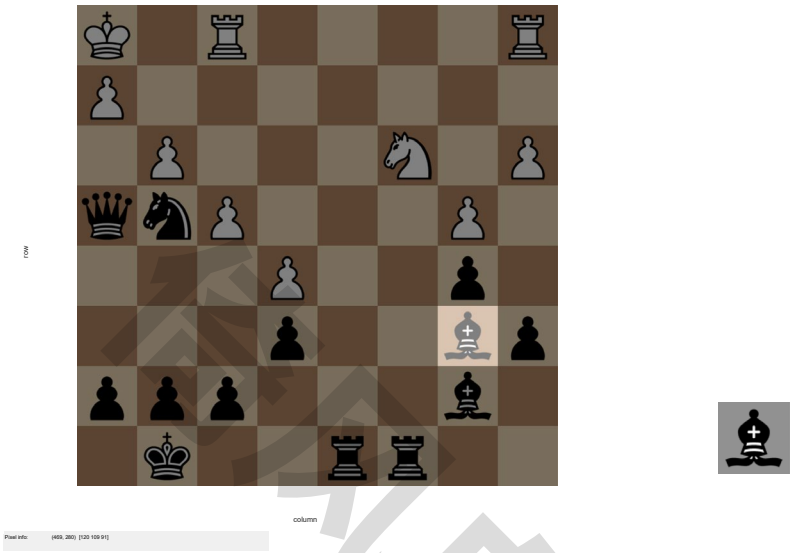
```
    col_start = col * board_size + 1;

    row_end = row_start + board_size - 1;
    col_end = col_start + board_size - 1;

    % Crop region starting at (100,120)
    im_crop = im(row_start : row_end,col_start:col_end,:);

    % convert unit8 into double to do matrix operation.
    im_gray = rgb2gray(im_crop);
    % 在原棋盘上显示选中的区域
    sig = zeros(size(im));
    sig(row_start : row_end,col_start:col_end,:,1) = 255;
    sig = uint8(sig);
   %虽然上面赋值是在 0~255，但在设 sig 数组时的默认类型为 double,所以要用 uint8 函数转为 uint8type
    pos = imlincomb(0.5,im,0.5,sig,'uint8');
    figure
    imshow(pos)
    xlabel('column')
    ylabel('row')
    impixelinfo;
    % Detemien how many zero(white) values in the image
    n = nnz(im_gray)
    % returns the number of nonzero elements in matrix X.
    m = numel(im_gray)
    % returns the number of elements, n, in array A, equivalent to prod(size(A)).

    if all(im_gray(:) == im_gray(1))
        disp('Empty board');
    else
        white_pixel = (n/m) * 100%value in (0,1]

        if white_pixel > 85
            disp('white piece');%当非白区占大多数，则认为为白棋
        else
            disp('black piece');
        end
    end

    % Display the croped image
    figure
    imshow(im_gray)
```

```
55. end
```

调用函数

```
1.  row = 5;
2.  column = 6;
3.  im_board = board_position(row,column);
```

输出结果如下



该棋子被判别为黑棋

Tutorial Activity 2: Spot the Difference !

1.函数列表

- **`bwlabel`**
  L = bwlabel(BW,conn) returns a label matrix, where conn specifies the connectivity.
- **`imbinarize`**
  BW = imbinarize(I,T):creates a binary image from image I using the threshold value T. T can be a global image threshold, specified as a scalar luminance value, or a locally adaptive threshold, specified as a matrix of luminance values.
- **`imshowpair`**
  obj = imshowpair(___,method):uses the visualization method specified by method.method 'montage' means that places A and B next to each other in the same image.
- **`rectangle('Position',pos)`**
  creates a rectangle in 2-D coordinates. Specify pos as a four-element vector of the form [x y w h] in data units. The x and y elements determine the location and the w and h elements determine the size. The function plots into the current axes without clearing existing content from the axes.

- **regionprops**

  Measure properties of image regions, properties we use is list as follow.

'BoundingBox'	Position and size of the smallest box containing the region, returned as a 1-by-(2*Q) vector. The first Q elements are the coordinates of the minimum corner of the box. The second Q elements are the size of the box along each dimension. For example, a 2-D bounding box with value [5.5 8.5 11 14] indicates that the (x,y) coordinate of the top-left corner of the box is (5.5, 8.5), the horizontal width of the box is 11 pixels, and the vertical height of the box is 14 pixels.

2.找不同

（1）思路分析

1.想要找出两张图的不同之处，可以逐个对比像素点的 RGB 值，并将其转换为灰度值以便后续计算，从而得到灰度值差值矩阵。进一步处理矩阵，先设定一个合适的阈值，若矩阵中的差值元素超过阈值，则认为两处不同。最后利用 `imlincomb` 函数将原图与差值图叠加，得到“找不同”游戏的答案。

（2）代码及运行结果

```
%% Tutorial Activity 2: Spot the Difference !
clc;close all;clear all;
f = imread('spot_the_difference.png');
[M,N,D] = size(f);
im1 = f(:,1:N/2,:);
im2 = f(:,N/2+1:end,:);

im_diff=im1-im2;
im_diff=rgb2gray(im_diff);%change into grayscale,(in range double[0,255])
im_diff=im_diff>40;%要转换为 double 类型进行计算，如用 uint8，可能数据溢出
im_diff=cat(3,im_diff*255,zeros(size(im_diff)),zeros(size(im_diff)));

%Concatenate arrays,which 3 is number of dimension.
im_diff = uint8(im_diff);%change into uint8scale
im_diff1 = im_diff;
im_diff = imlincomb(0.5,im1,10,im_diff,'uint8');
figure;
subplot(1,3,1);imshow(im1);
subplot(1,3,2);imshow(im2);
subplot(1,3,3);imshow(im_diff)
% figure;imshow(im_diff1)
% d =  bwconncomp(im_diff1)
```



3.计算连通区域

（1）思路分析

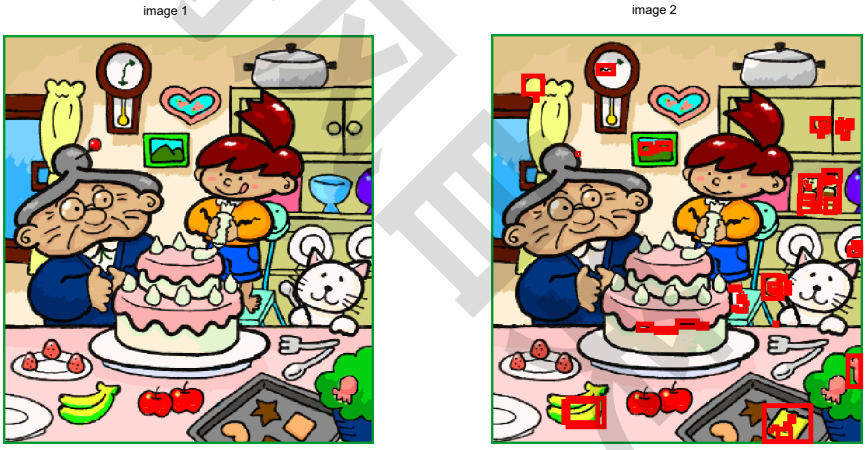
我们想借助 bwlabel 或者 bwconncomp 函数获取连通区域的属性，以更好地标记处两张图像的不同点，为了计算连通区域，我们需要将 RGB 图转换为二值图。先仿照之前的做法，计算出两图像像素点 RGB 差值，再转换为灰度值，借助 graythresh 算出合理的阈值，并以此为标准，再将灰度值进一步转换为二值图。采用 bwlabel、regionprops 计算连通区域及其属性，最后借助 **rectangle 函数绘图。**

（2）代码及运行结果

```
clc;close all;clear all;
f = imread('spot_the_difference.png');
[M,N,D] = size(f);
img1 = f(:,1:N/2,:);
img2 = f(:,N/2+1:end,:);

% calculate the size
img1_size = size(img1);
img2_size = size(img2);

% 防止图像不匹配
if(~isequal(img1_size, img2_size))
    error('The two images must be the same size and shade of colour.');
end

% %计算图像差值
% kep_kulonbseg1 = img2 - img1;
% kep_kulonbseg2 = img1 - img2;
```

```
% %比如图像 a 和 b，做 a（3,3,2）-b（3,3，1），如果相减是个负数的话 matla 会自动处理为 0,
% kep_kulonbseg = kep_kulonbseg1 + kep_kulonbseg2;

% 计算图像差值
im_diff = img1 - img2;
figure;imshow(im_diff)
% 转换为灰度值
gray_diff = rgb2gray(im_diff);

%使用 graythresh 计算阈值。阈值归一化至范围 [0, 1]。
treshold = graythresh(gray_diff);
% treshold = 0.4;%也可以自己设定阈值
%使用阈值将图像转换为二值图像。
BW = imbinarize(gray_diff,treshold);
figure;imshow(BW)
%在二值图像旁边显示原始图像。
figure;imshowpair(gray_diff,BW,'montage')
connect_area = bwlabel(BW,8);
connect_num =  bwconncomp(BW)
%对二维二值图像中的连通分量进行标注
%详见参考资料：https://ww2.mathworks.cn/help/images/ref/bwlabel.html?searchHighlight=bwlabel&s_tid=srchtitle#bupqqy6-1-n
stat = regionprops(connect_area, 'BoundingBox');
% 获取连通区域的属性
figure
subplot(1,2,1),
imshow(img1);
title('image 1');

%用矩形框出连通区域
subplot(1,2,2),
imshow(img2);
title('image 2');
for k = 1:size(stat)
    rectangle('Position', stat(k).BoundingBox, 'EdgeColor','r','LineWidth',2);
end
```

差值灰度图（左）及二值图（右）



运行程序可知，共有 68 个连通区域。

```
connect_num =
```

包含以下字段的 struct:

```
  Connectivity: 8
     ImageSize: [386 350]
    NumObjects: 68
  PixelIdxList: {1×68 cell}
```

4.图像的膨胀

（1）函数列表

- **imdilate**

  J = imdilate(I,SE) dilates the grayscale, binary, or packed binary image I, returning the dilated image, J. SE is a structuring element object or array of structuring element objects, returned by the strel or offsetstrel functions.

- **strel**

  SE = strel('disk',r,n) creates a disk-shaped structuring element, where r specifies the radius and n specifies the number of line structuring elements used to approximate the disk shape. Morphological operations using disk approximations run much faster when the structuring element uses approximations.

（2）思路分析

不足：在上述计算连通区域的过程中，由于计算两图 RGB 差值矩阵时，会存在断点。故基于 8 连通计算连通区域时，一个“不同点”（蛋糕奶油花纹）可能存在多个连通区域，因而被多次标记，不符合预期结果。后续可采用其他算法将靠近的连通区域判定为一个“不同点”，从而实现一个矩形框对应一个“不同点”

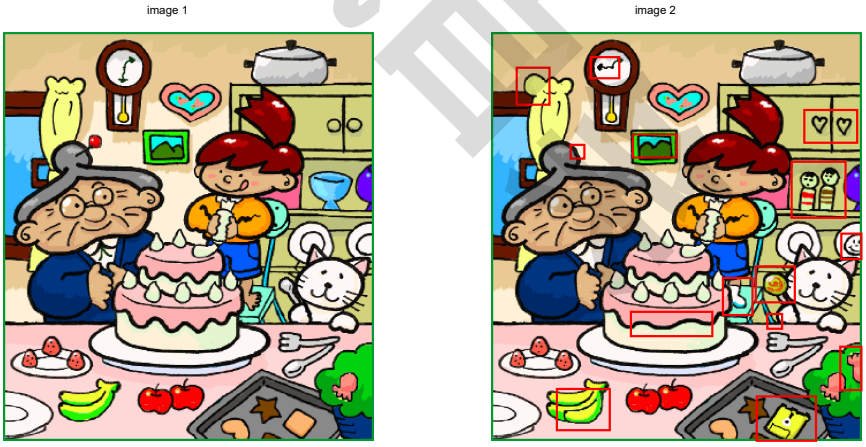
蛋糕奶油花纹处的差值二值图

改进：采用 strel、imdilate 函数对二值图进行膨胀，使相邻的连通区域合并，效果如下

（3）代码与运行结果

```
%用 imdilate 函数对二值图进行膨胀，修正连通区域。
BW_inflation= imdilate(BW,strel('disk',7));
figure;imshow(BW_inflation);
connect_area = bwlabel(BW_inflation,8);
connect_num =  bwconncomp(BW_inflation)
%对二维二值图像中的连通分量进行标注
```

```
%详见参考资料：https://ww2.mathworks.cn/help/images/ref/bwlabel.html?searchHighlight=bwlabel&s_tid=srchtitle#bupqqy6-1-n
stat = regionprops(connect_area, 'BoundingBox');
% 获取连通区域的属性
figure
subplot(1,2,1),
imshow(img1);
title('image 1');

%用矩形框出连通区域
subplot(1,2,2),
imshow(img2);
title('image 2');
for k = 1:size(stat)
    rectangle('Position', stat(k).BoundingBox, 'EdgeColor','r','LineWidth',1);
end
```



参考资料：

1.matlab 图像类型转换以及 uint8、double、im2double、im2uint8 和 mat2gray 等说明_无鞋童鞋的博客-CSDN 博客

2.https://github.com/Budincsevity/Spot-the-difference/blob/master/spotTheDifference.m

3.Matlab 中 imregionalmax 函数 和 bwconncomp 函数的使用_学生的博客-CSDN 博客
4.图像处理-图片找不同 Find the differences of the two images #Matlab 下调用 Python 接口 #SIFT 算法应用_夏普通-CSDN 博客