实验报告

16340282 袁之浩

相关检测

原理：利用两幅图像的相关值来解决图像匹配的问题，因为把被匹配图像作为 kernel，然后需要检测的图像作为 pattern，相关可以用来检测和定位一个由 kernel 表示的 pattern，被检测处 pattern 响应最强。为了避免图像本身亮度值造成的干扰，需要将图像归一化。

计算公式：$$G(i,j)=\frac{\sum_{u=-k}^{k}\sum_{v=-l}^{l}H(u,v)F(i+u,j+v)}{\sum_{u=-k}^{k}\sum_{v=-l}^{l}F^2(i-u,j-v)}$$，其中 H(u,v) 为匹配模板， 而分母为图像内容。

算法：首先将 car 和 wheel 图像读入，保存在矩阵中。

```
car = imread('car.png');
mask = imread('wheel.png');
```

然后将 car 矩阵的边缘扩充补 0，扩充的大小就是目标图像的大小-1。先声明一个全 0 的扩充矩阵，然后在对应位置改为 car 的像素值。

```
% car_ext为对car进行边缘补零后的图像
car_ext = zeros(height1+height2-1,width1+width2-1);
h2_2 = floor(height2/2);    % 模板的半高
w2_2 = floor(width2/2);    % 模板的半宽
m1 = h2_2+1;       % 原图像car在边缘补零后的图像car_ext中的起始行位置
m2 = h2_2+height1;
n1 = w2_2+1;       % 原图像car在边缘补零后的图像car_ext中的起始列位置
n2 = w2_2+width1;
car_ext(m1:m2,n1:n2) = car;
```

然后根据上述公式进行计算。对 car 中的每一个像素，获得它的邻域，邻域的大小就是要匹配的图像的大小，然后将模板与原图像对应位置相乘后累加，同时计算原像素的平方用于归一化。

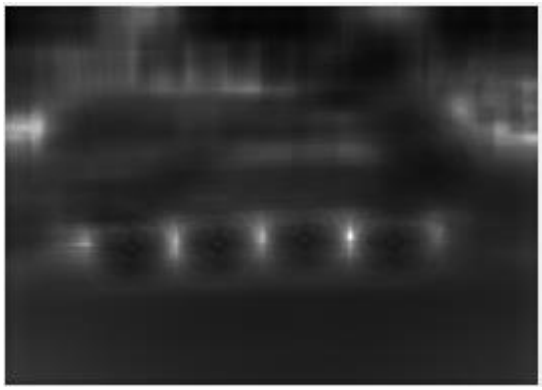
```
sum1 = 0;
sum2 = 0;
m(i-h2_2:i+h2_2,j-w2_2:j+w2_2) = mask;
for u = i-h2_2:i+h2_2
    for v = j-w2_2:j+w2_2
        sum1 = sum1+car_ext(u,v)*m(u,v);
        sum2 = sum2+car_ext(u,v)^2;  % su
    end
end
g(i,j) = sum1/sum2;    % 归一化
```

把相关矩阵去除边缘的 0 后，归一化到 0-255，然后找出最大值，并求出坐标。

```
最大值坐标：
z =

   116   173
```

相关矩阵，可以看到最亮的地方就是我们要匹配的图像。



中值滤波

原理：中值滤波的特点即是首先确定一个以某个像素为中心点的邻域，然后将邻域中各像素的灰度值排序，取其中间值作为中心像素灰度的新值，这里领域被称为窗口，当窗口移

动时，利用中值滤波可以对图像进行平滑处理。其算法简单，时间复杂度低，对于椒盐噪声有很好的效果。

算法：首先随机生成两个和原图一样大的矩阵

```
f0=imread('sport car.pgm');
[height,width]=size(f0);
t1=255*rand([height,width]);
t2=255*rand([height,width]);
```

根据如下方式产生椒盐噪声图像：

$$f(x,y)=\begin{cases}255 & \text{if } f_0(x,y)>t_1(x,y)\\ 0 & \text{if } f_0(x,y)<t_2(x,y)\\ f_0(x,y) & \text{otherwise}\end{cases}$$

即如果原像素值大于两个函数值，就取 255，小于就取 0，在两者之间则维持不变。

```
f=size(f0);
for i=1:height
    for j=1:width
        if f0(i,j) > max(t1(i,j),t2(i,j))
            f(i,j)=255;
        elseif f0(i,j) < min(t1(i,j),t2(i,j))
            f(i,j)=0;
        else
            f(i,j)=f0(i,j);
        end
    end
end
```

采用 3*3 窗口实现中值滤波，这里不需要扩充，直接从第（2，2）个像素值遍历到第（height-1，weight-1），将像素值的邻域像素变成一个行矩阵，然后取中值。

```
f1=f;
for i=2:height-1
    for j=2:width-1
        c=f(i-1:i+1,j-1:j+1);    %3*3邻域
        e=c(1,:);        %c矩阵的第一行
        for u=2:3
            e=[e,c(u,:)];     %将c矩阵变为一个行矩阵
        end
        m=median(e);     %取中值
        f1(i,j)=m;
    end
end
```

显示原图像、椒盐噪声图像、中值滤波图像，并和采用 Matlab ‘medfilt2’

的结果做比较。