一、

1.

a 为 B 通道；b 为 G 通道；c 为灰度图；d 为 R 通道。由于某个通道的灰度图像中的明暗对应该通道色的明暗，因而表达出该色光在整体图像上的分布情况。首先，原图海绵宝宝的脸大面积为黄色，而黄色的补色为蓝色，在 B 通道中脸会呈暗色，可以先确定 a 为 B 通道；接着，原图海绵宝宝系着红色的领结，故在 R 通道中这部分会呈现亮色，可以确定 d 为 R 通道；然后，还是看红色领结，在 G 通道中红色领结部位应该呈现暗色，故 b 为 G 通道；由于灰度图在 R 通道占比 0.299，在 G 通道占比 0.587，故灰度图在红色领结部位依然呈暗色，但相比 G 通道图会更亮一些。图 b，c 相似，正是因为 b 为 G 通道，c 为灰度图，并且灰度图在 G 通道上占比 0.587。

2.

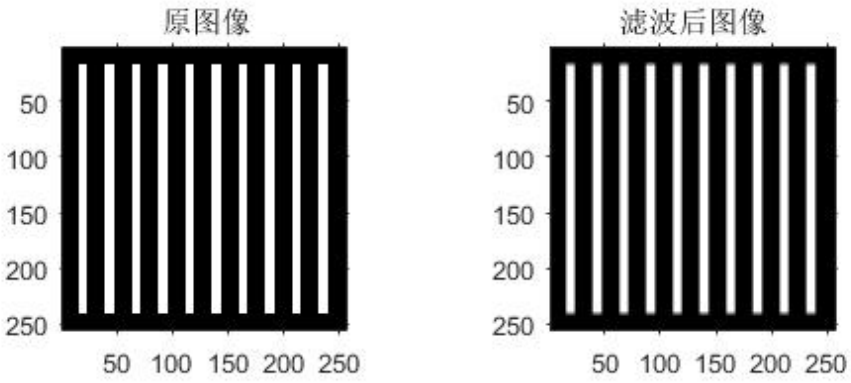
图 b。将原图从 RGB 彩色空间转换到 HSI 彩色空间后，原色、二次色都是按 120° 分隔的，与红轴的 0° 角指定为 0 色调，从这里开始色调逆时针增长。对 H 通道加上 60° 后，红色领结部分将变为黄色，脸部黄色将变为绿色，原图中脸部的绿色小圈将变为青色，蓝色眼珠变为深红色，白色眼睛和牙齿依然为白色，综上所述，只有图 b 符合。

二、

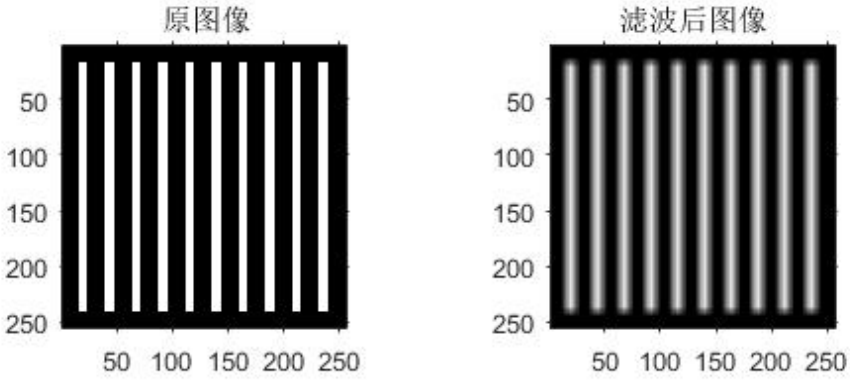
2.2

1.

3*3



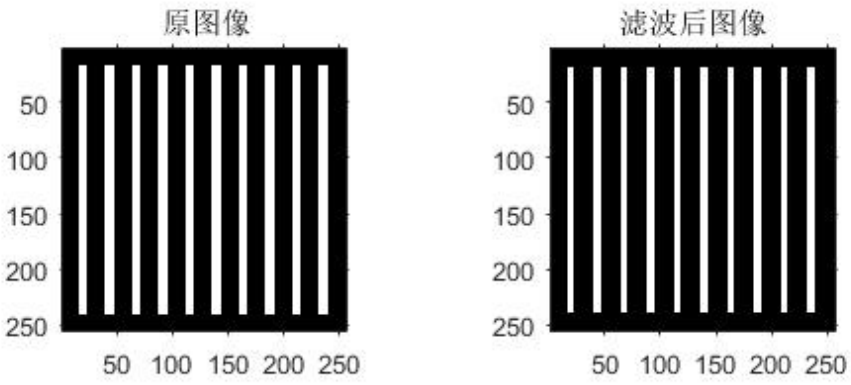
9*9



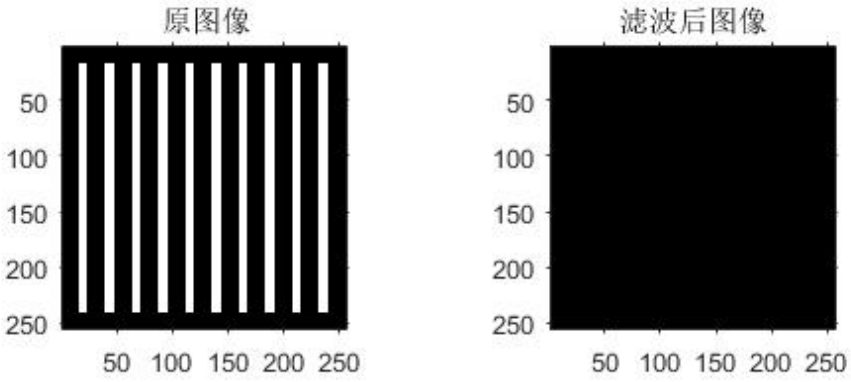
特点：滤波后图像变得平滑，而 9*9 滤波结果显得更加模糊，白条稍微变窄，变矮，白条边缘变灰。

2.

3*3



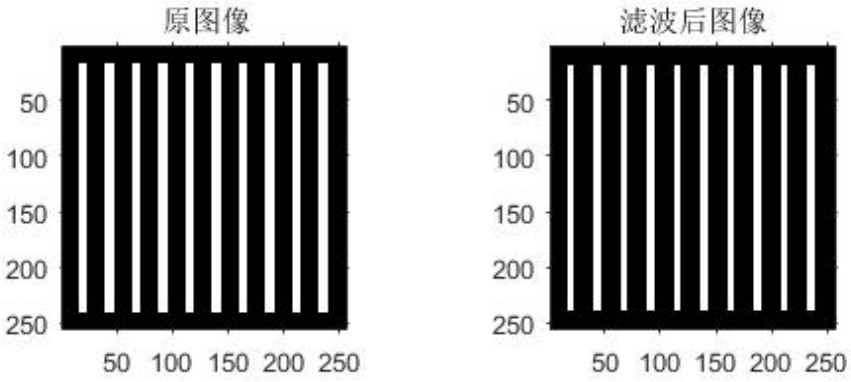
9*9



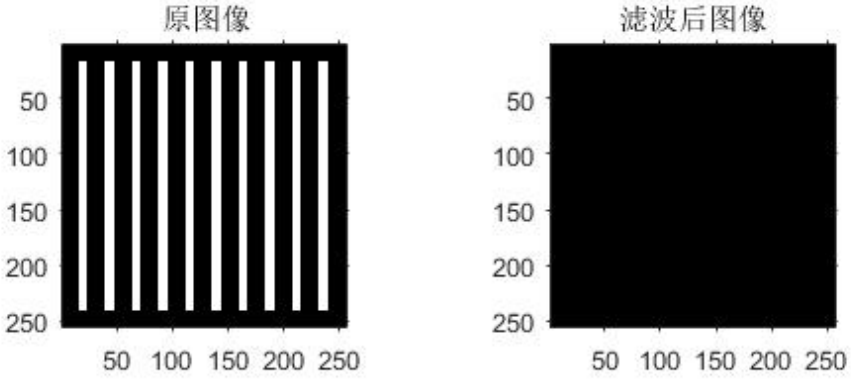
特点：3*3 滤波效果稍微变得平滑一些，9*9 滤波，由于区域过大，导致每个区域含有像素为 0 的点，最后造成目标像素全部被赋值为 0。

3.

3*3



9*9



特点：3*3 滤波图像变得平滑一些，白条变短变窄；而 9*9 滤波，由于区域过大，导致每个区域含有像素为 0 的点，最后造成目标像素全部被赋值为 0，最后得到全黑图像。

2.3

1. 高斯噪声生成部分代码：

mean 为均值，std 为标准差。

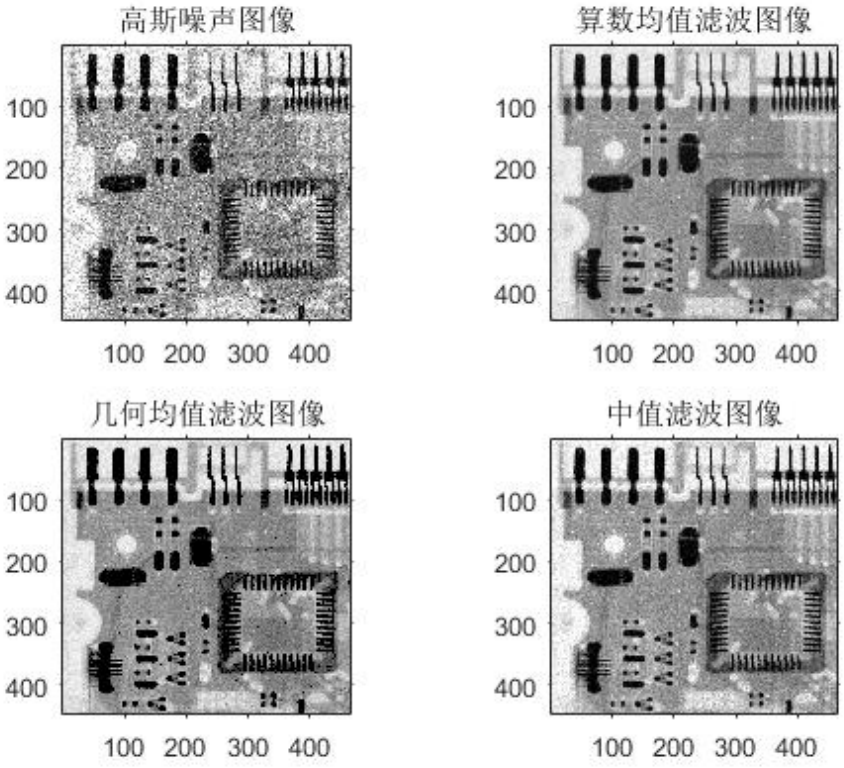
```
g_noise = zeros(M,N);
gaussian = mean + std .* randn(M,N);
for i = 1:M
    for j = 1:N
        g_noise(i,j) = img1(i,j) + gaussian(i,j);
    end
end
g_noise = uint8(g_noise);
```

椒盐噪声生成部分代码：

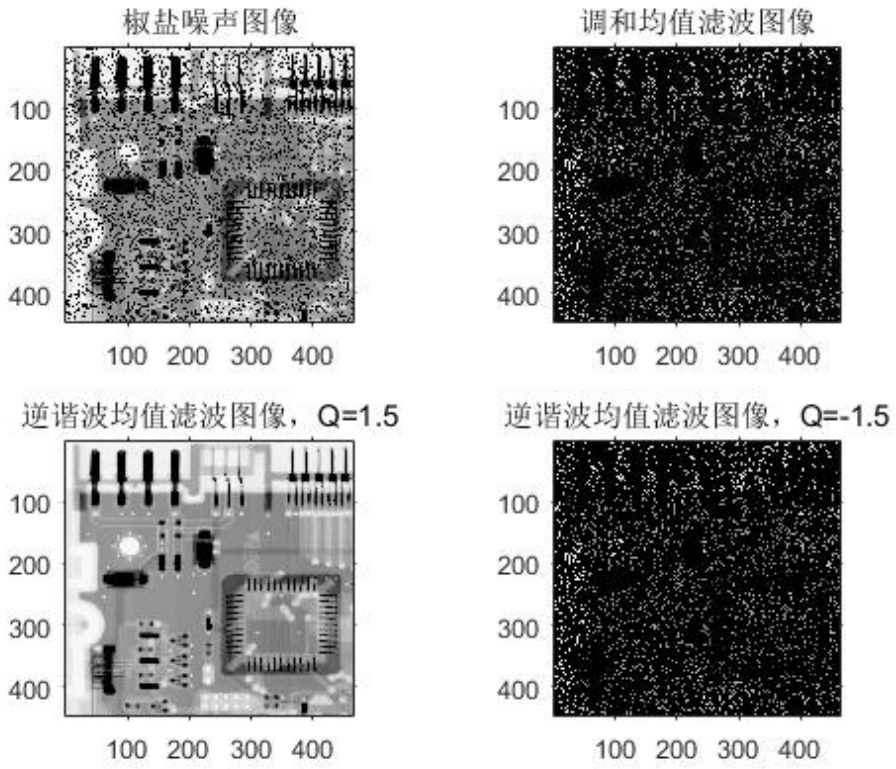
a 为椒噪声概率，b 为盐噪声概率。

```
a = 0;
b = 0.2;
x = rand(M,N);
salt_noise = img;
salt_noise(find(x<a)) = 0;
salt_noise(find(x > a & x<(a+b))) = 1;
salt_noise = uint8(salt_noise);
```

2.滤波器大小均选取 3*3；



结果比较：算数均值滤波器处理后降低了噪声，但同时也对图像造成一定的模糊；
几何均值滤波器同算数均值滤波器一样降噪明显，但这种处理丢失的图像细节更少；
中值滤波处理效果相比以上两种线性滤波器，得到的结果模糊更少。
3.滤波器大小均选取 3*3；



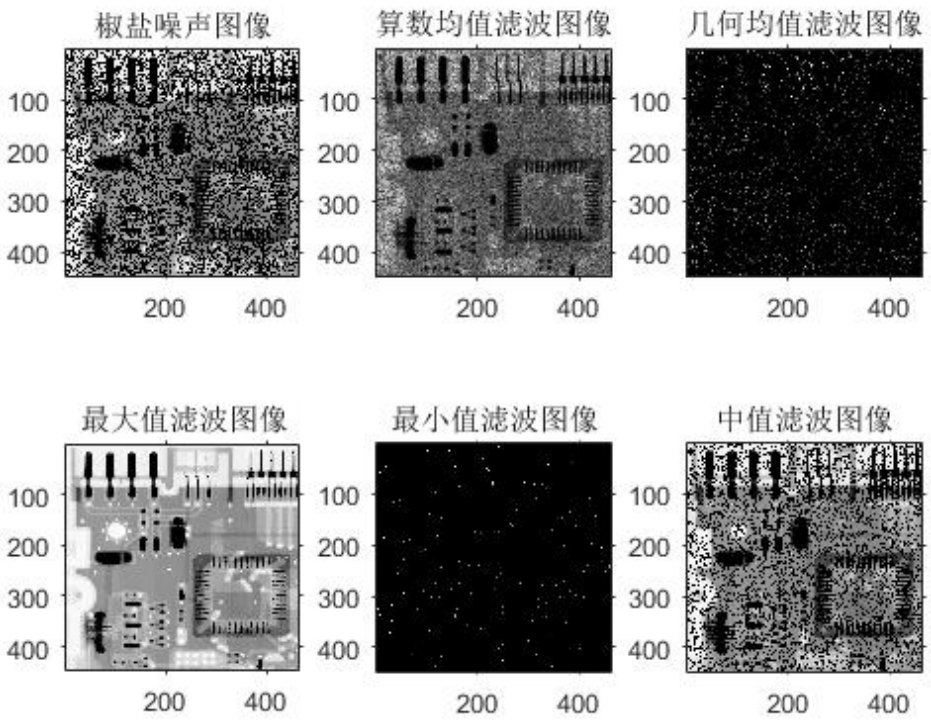
通过分析逆谐波均值滤波器的表达式即可得知 Q 值的正负对滤波结果的影响。

$$\hat{f}(x,y)=\frac{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q+1}}{\sum_{(s,t)\in S_{xy}} g(s,t)^{Q}}$$

；

当 Q 值为正时，该滤波器消除胡椒噪声；当 Q 值为负时，该滤波器消除盐粒噪声。倘若 Q 值选取正值去滤波被盐噪声污染的图像，从表达式可知，对任意一点盐噪声 f(x,y)，经运算后得到的值依然比周围像素值高出不少，无法去除盐粒噪声；同理，若 Q 取负值去消除胡椒噪声，噪声处经运算后得到的值跟原值相差不大，依旧比周围像素值小得多，同样也无法达到预期效果，甚至一些本不是噪声点经处理后也变成了噪声点。

4.滤波器大小均选取 3*3；



由以上图示结果可看出，最大值滤波效果最好，几何均值滤波、最小值均值滤波出现了灾难性结果。最大值滤波通过选取区域内最大值赋值给区域中心像素，这样处理会完全消除胡椒噪声，但对盐粒噪声效果不太好，所以结果图像偏亮；相反，最小值滤波选取最小值，因而整体图像都变暗，效果不好。算数均值滤波、中值滤波都一定程度的降低了噪声，也稍微平滑了图像，但降噪效果不明显；几何均值滤波效果极差，是因为图像噪声为椒盐噪声，图像中存在许多像素值为 0 的点，从而大部分区域内像素点的乘积会变为 0，所以效果不好。

5.

算数均值滤波：

通过写出逆谐波均值滤波表达式，然后 Q 值取 0；

```
c=temp1(i:i+(n-1),j:j+(n-1));
temp2(i+(n-1)/2,j+(n-1)/2) = sum(c(:).^(Q+1))/sum(c(:).^(Q));
```

几何均值滤波：

区域内像素点乘积在开方赋值给区域中心像素；

```
temp3(i+(n-1)/2,j+(n-1)/2) = prod(c(:).^(1/numel(c)));
```

调和均值滤波：

通过写出逆谐波均值滤波表达式，然后 Q 值取-1；

```
temp2(i+(n-1)/2,j+(n-1)/2) = sum(c(:).^(Q+1))/sum(c(:).^(Q));
```

逆谐波均值滤波：

直接写出逆谐波均值滤波表达式，根据噪声类型适当选取 Q 值；

```
temp2(i+(n-1)/2,j+(n-1)/2) = sum(c(:).^(Q+1))/sum(c(:).^(Q));
```

最大值滤波：

选取区域内最大值并赋值给区域中心像素；

```
temp4(i+(n-1)/2,j+(n-1)/2) = max(c(:));
```

最小值滤波：

选取区域内最大值并赋值给区域中心像素；

```
temp5(i+(n-1)/2,j+(n-1)/2) = min(c(:));
```

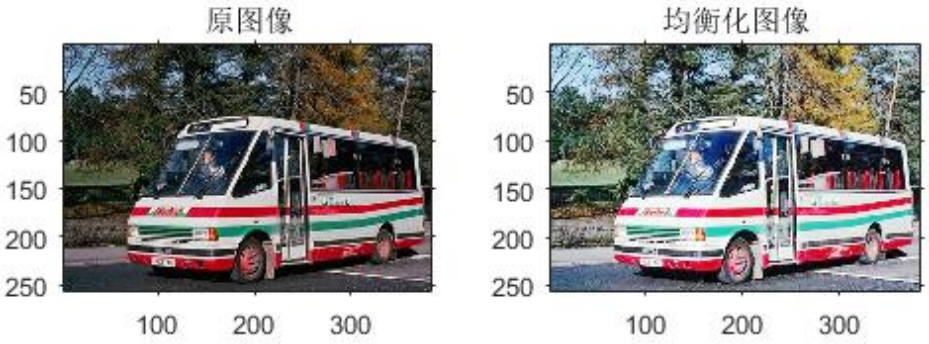
中值滤波：

选取区域内中值并赋值给区域中心像素；

```
temp = sort(c(:));
temp6(i+(n-1)/2,j+(n-1)/2) = temp((numel(temp)+1)/2);
```

2.4

1.



2.

3.

4.

第一种方法是分别对 R、G、B 三个通道进行直方图均衡化，然后将处理后的三通道重构成一张 RGB 图；第二种方法是对 R、G、B 三个通道直方图取平均值得到一个平均直方图，对这个平均直方图做均衡化再分别映射到 R、G、B 三个通道，再重构一张 RGB 图；第三种方法是将输入图片先转换到 HSI 色彩空间，对强度通道进行直方图均衡化，再将处理后的结果转换到 RGB 色彩空间。前两种方法类似，都是在 RGB 彩色空间内完成直方图均衡化的，根据图示结果可以看到虽然的确有将原图中的阴暗部分变得明亮起来，但是颜色的失真也是比较严重的。在均衡化过程中不仅改变了亮度，也改变了彩色，产生了不正确的彩色。通过第三种

在 HSI 彩色空间均衡化方法得到的结果图像效果是比较好的，整个图像都有效的加亮了，而彩色本身（色调）是不变的。