滤波器

电子信息工程系

袁羽

目录

CONTENTS

一 图像噪声

图像噪声是指存在于图像数据中的不必要的或多余的干扰信息。噪声的存在严重影响了图像的质量，因此在图像增强处理和分类处理之前，必须予以纠正。

图像系统中的噪声来自多方面 ，有电子元器件 ，如电阻引起的热噪声；真空器件引起的散粒噪声和闪烁噪声；面结型晶体管产生的颗粒噪声和噪声；场效应管的沟道热噪声 ；光电管的光量子噪声和电子起伏噪声；摄象管引起的各种噪声等等。由这些元器件组成各种电子线路以及构成的设备又将使这些噪声产生不同的变换而形成局部线路和设备的噪声；另外还有光学现象所产生的图像光学噪声。

一 图像噪声

图像中可能会出现这样一种像素，该像素与周围像素的差别非常大，导致从视觉上就能看出该像素无法与周围像素组成可识别的图像信息，降低了整个图像的质量。这种“格格不入”的像素就是图像的噪声。如果图像中的噪声都是随机的纯黑像素或者纯白像素，这样的噪声称作“椒盐噪声”或“盐噪声”。

一 图像噪声

在尽量保留原图像信息的情况下，去除图像内噪声、降低细节层次信息等一系列过程，叫作图像的平滑处理（或图像的模糊处理）。实现平滑处理最常用的工具就是滤波器。通过调节滤波器的参数，可以控制图像的平滑程度。OpenCV提供了种类丰富的滤波器，每种滤波器使用的算法均不同，但都能对图像中的像素值进行微调，让图像呈现平滑效果。

二 均值滤波器

以一个像素为核心，其周围像素可以组成一个n行n列（简称n×n）的矩阵，这样的矩阵结构在滤波操作中被称为“滤波核”。矩阵的行、列数决定了滤波核的大小。

均值滤波器（也称为低通滤波器）可以把图像中的每一个像素都当成滤波核的核心，然后计算核内所有像素的平均值，最后让核心像素值等于这个平均值。

二 均值滤波器

采用邻域平均法的均值滤波器非常适用于去除通过扫描得到的图像中的颗粒噪声。领域平均法有力地抑制了噪声,同时也由于平均而引起了模糊现象,模糊程度与领域半径成正比。

69	42	123	145	120	172	96
45	99	163	168	115	48	87
89	100	159	82	86	95	152
77	121	165	180	188	204	77
85	152	154	55	68	210	56
63	142	143	177	180	158	102
65	63	88	92	95	99	112

69	42	123	145	120	172	96
45	99	163	168	115	48	87
89	100	159	82	86	95	152
77	121	165	180	188	204	77
85	152	154	55	68	210	56
63	142	143	177	180	158	102
65	63	88	92	95	99	112

二 均值滤波器

OpenCV将均值滤波器封装成blur()方法，其语法如下：

dst = cv2.blur(src, ksize, anchor, borderType)

参数说明：

src：被处理的图像。

ksize：滤波核大小，其格式为(高度，宽度)，建议使用如(3，3)、(5，5)、(7，7)等宽、高相等的奇数边长。滤波核越大，处理之后的图像就越模糊。

anchor：可选参数，滤波核的锚点，建议采用默认值，可以自动计算锚点。

borderType：可选参数，边界样式，建议采用默认值。

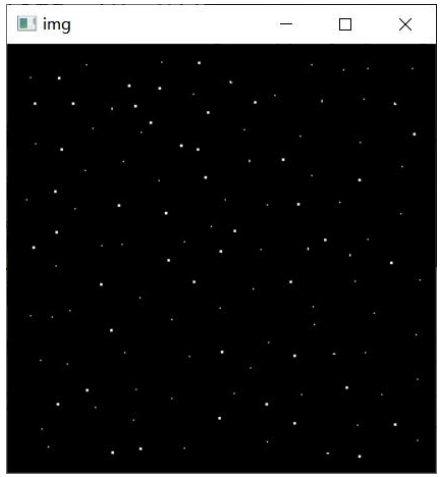
返回值说明：

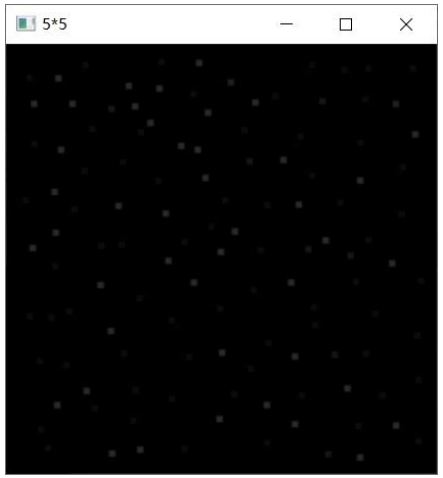
dst：经过均值滤波处理之后的图像。

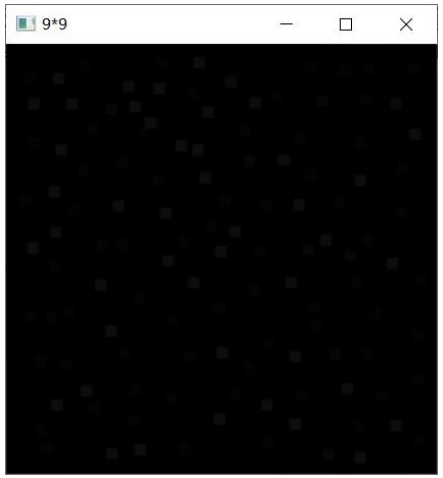
例：对图像进行均值滤波操作：分别使用大小为3×3、5×5和9×9的滤波核对噪声图像进行均值滤波操作。

```
import cv2
img = cv2.imread("noise.jpg") # 读取原图
dst1 = cv2.blur(img, (3, 3)) # 使用大小为3*3的滤波核进行均值滤波
dst2 = cv2.blur(img, (5, 5)) # 使用大小为5*5的滤波核进行均值滤波
dst3 = cv2.blur(img, (9, 9)) # 使用大小为9*9的滤波核进行均值滤波
cv2.imshow("img", img) # 显示原图
cv2.imshow("3*3", dst1) # 显示滤波效果
cv2.imshow("5*5", dst2)
cv2.imshow("9*9", dst3)
cv2.waitKey() # 按下任何键盘按键后
cv2.destroyAllWindows() # 释放所有窗体
```

例：对图像进行均值滤波操作：分别使用大小为3×3、5×5和9×9的滤波核对噪声图像进行均值滤波操作。







三 中值滤波器

中值滤波器的原理与均值滤波器非常相似，唯一的不同就是不计算像素的平均值，而是将所有像素值排序，把最中间的像素值取出，赋值给核心像素。

以下滤波核大小为3×3，核心像素值是9，周围像素值都为130～200。将核内所有像素值按升序排列，9个像素值排成一行，最中间位置为第5个位置，这个位置的像素值为162。不需再做任何计算，直接把162赋值给核心像素，其颜色就与周围颜色差别不大，图像就变得平滑了。

将滤波核像素升序排列取中值

三 中值滤波器

OpenCV将中值滤波器封装成medianBlur()方法，其语法如下：

```
dst = cv2.medianBlur(src, ksize)
```

参数说明：

src：被处理的图像。

ksize：滤波核的边长，必须是大于1的奇数，如3、5、7等。该方法根据此边长自动创建一个正方形的滤波核。

返回值说明：

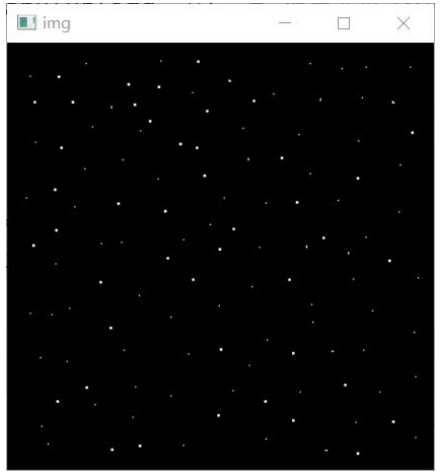
dst：经过中值滤波处理之后的图像。

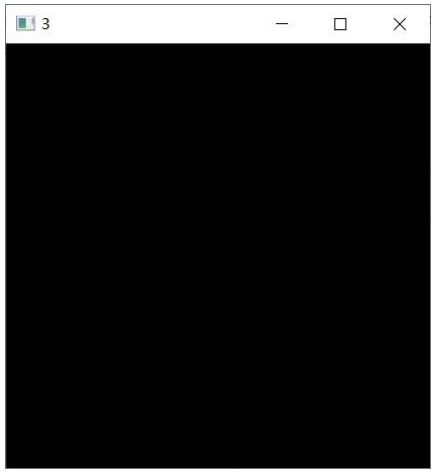
注意：中值滤波器的ksize参数是边长，而其他滤波器的ksize参数通常为（高，宽）。

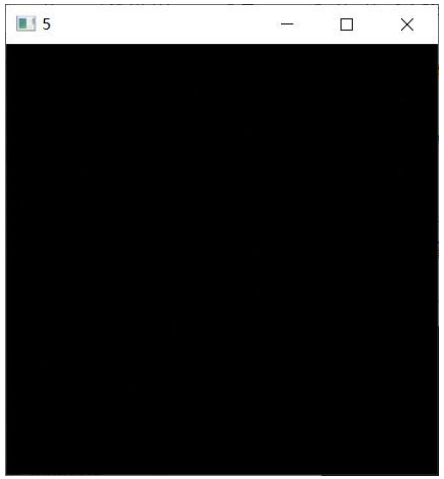
例：对图像进行中值滤波操作：分别使用边长为3、5、9的滤波核对噪声图像进行中值滤波操作。

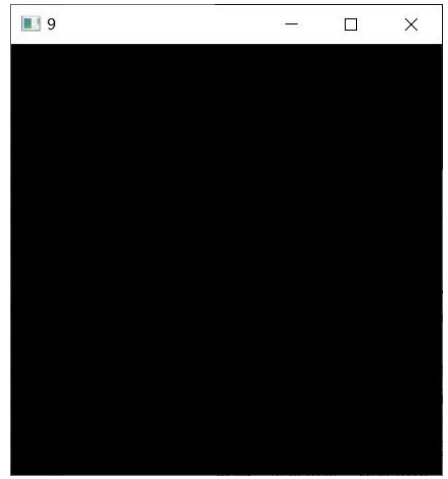
```
import cv2
img = cv2.imread("noise.jpg") # 读取原图
dst1 = cv2.medianBlur(img, 3) # 使用宽度为3的滤波核进行中值滤波
dst2 = cv2.medianBlur(img, 5) # 使用宽度为5的滤波核进行中值滤波
dst3 = cv2.medianBlur(img, 9) # 使用宽度为9的滤波核进行中值滤波
cv2.imshow("img", img) # 显示原图
cv2.imshow("3", dst1) # 显示滤波效果
cv2.imshow("5", dst2)
cv2.imshow("9", dst3)
cv2.waitKey() # 按下任何键盘按键后
cv2.destroyAllWindows() # 释放所有窗体
```

例：对图像进行中值滤波操作：分别使用边长为3、5、9的滤波核对噪声图像进行中值滤波操作。







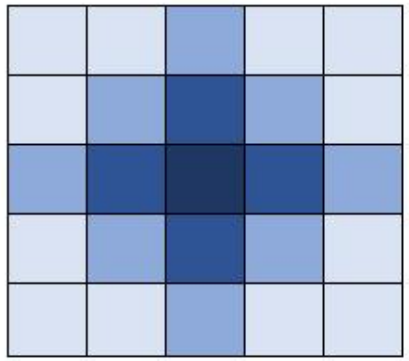


滤波核的边长越长，处理之后的图像就越模糊。中值滤波处理的图像会比均值滤波处理的图像丢失更多细节，而对于椒盐噪声，中值滤波的处理效果更好。

四 高斯滤波器

高斯滤波也被称为高斯模糊或高斯平滑，是目前应用最广泛的平滑处理算法。高斯滤波可以很好地在降低图片噪声、细节层次的同时保留更多的图像信息，经过处理的图像呈现“磨砂玻璃”的滤镜效果。

进行均值滤波处理时，核心周围每个像素的权重都是均等的，也就是每个像素都同样重要，所以计算平均值即可。但在高斯滤波中，越靠近核心的像素权重越大，越远离核心的像素权重越小。



四 高斯滤波器

高斯滤波的计算过程涉及卷积运算，会有一个与滤波核大小相等的卷积核。卷积核中保存的值就是核所覆盖区域的权重值。卷积核中所有权重值相加的结果为1。随着核大小、σ标准差的变化，卷积核中的值也会发生较大变化。

135*0.05+144*0.1+153*0.05+162*0.1+9*0.4+171*0.1+180*0.05+189*0.1+198*0.05=103.5

最后得到的这个结果四舍五入就是高斯滤波的计算结果，滤波核的核心像素值从35改为104。

四 高斯滤波器

OpenCV将高斯滤波器封装成了GaussianBlur()方法，其语法如下：

dst = cv2.GaussianBlur(src, ksize, sigmaX, sigmaY, borderType)

参数说明：

src：被处理的图像。

ksize：滤波核的大小，宽高必须是奇数，如(3, 3)、(5, 5)等。

sigmaX：卷积核水平方向的标准差。

sigmaY：卷积核垂直方向的标准差。修改sigmaX或sigmaY的值都可以改变卷积核中的权重比例。如果不知道如何设计这2个参数值，就直接把这2个参数的值写成0，该方法就会根据滤波核的大小自动计算合适的权重比例。

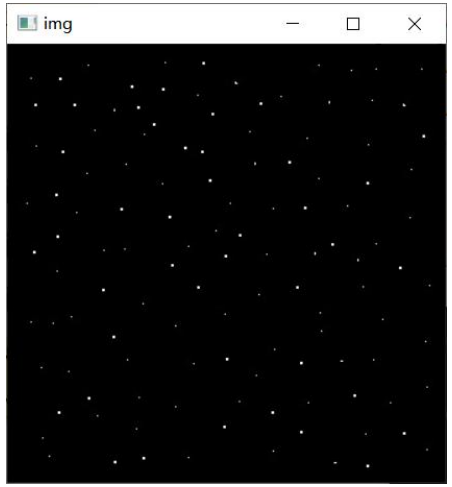
borderType：可选参数，边界样式，建议使用默认值。

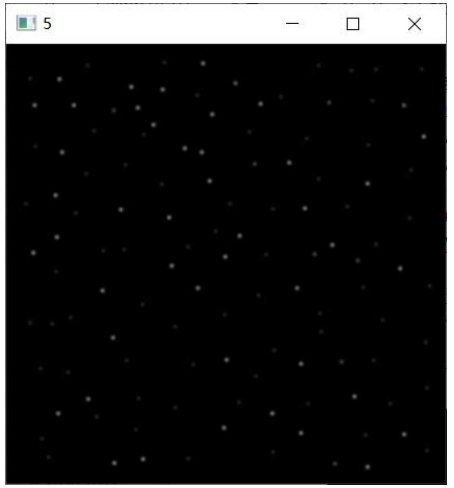
返回值说明：dst：经过高斯滤波处理之后的图像。

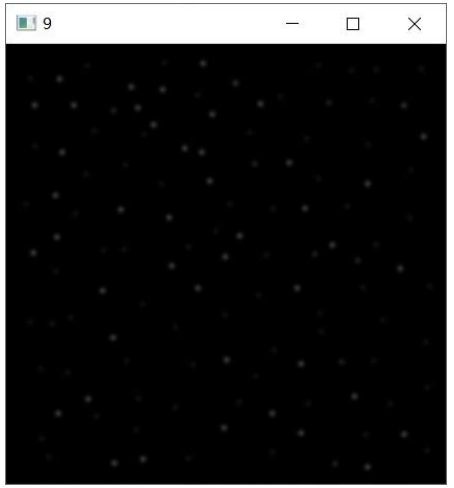
例：对图像进行高斯滤波操作：分别使用大小为5×5、9×9和15×15的滤波核对图像进行高斯滤波操作，水平方向和垂直方向的标准差参数值全部为0。

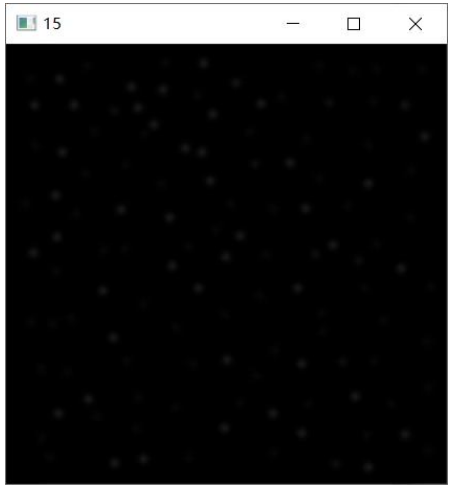
```
import cv2

img = cv2.imread("gaussi.png") # 读取原图

dst1 = cv2.GaussianBlur(img, (5, 5), 0, 0) # 使用大小为5*5的滤波核进行高斯滤波

dst2 = cv2.GaussianBlur(img, (9, 9), 0, 0) # 使用大小为9*9的滤波核进行高斯滤波

dst3 = cv2.GaussianBlur(img, (15, 15), 0, 0) # 使用大小为15*15的滤波核进行高斯滤波

cv2.imshow("img", img) # 显示原图

cv2.imshow("5", dst1) # 显示滤波效果

cv2.imshow("9", dst2)

cv2.imshow("15", dst3)

cv2.waitKey() # 按下任何键盘按键后

cv2.destroyAllWindows() # 释放所有窗体
```

例：对图像进行高斯滤波操作：分别使用大小为5×5、9×9和15×15的滤波核对图像进行高斯滤波操作，水平方向和垂直方向的标准差参数值全部为0。









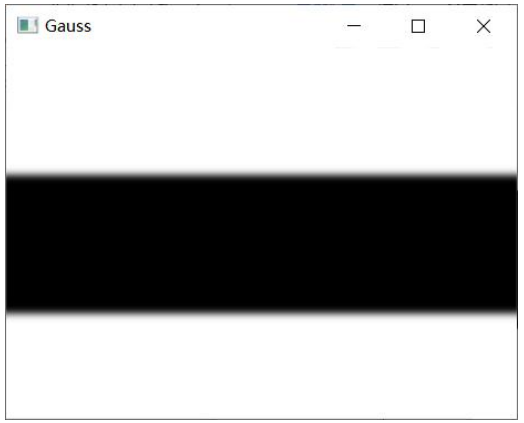
例：对图像进行高斯滤波操作：分别使用大小为5×5、9×9和15×15的滤波核对图像进行高斯滤波操作，水平方向和垂直方向的标准差参数值全部为0。

```
import cv2

img = cv2.imread("gaussi.png") # 读取原图

dst1 = cv2.GaussianBlur(img, (5, 5), 0, 0) # 使用大小为5*5的滤波核进行高斯滤波

dst2 = cv2.GaussianBlur(img, (9, 9), 0, 0) # 使用大小为9*9的滤波核进行高斯滤波

dst3 = cv2.GaussianBlur(img, (15, 15), 0, 0) # 使用大小为15*15的滤波核进行高斯滤波

cv2.imshow("img", img) # 显示原图

cv2.imshow("5", dst1) # 显示滤波效果

cv2.imshow("9", dst2)

cv2.imshow("15", dst3)

cv2.waitKey() # 按下任何键盘按键后

cv2.destroyAllWindows() # 释放所有窗体
```

五 双边滤波器

不管是均值滤波、中值滤波还是高斯滤波，都会使整幅图像变得平滑，图像中的边界会变得模糊不清。双边滤波是一种在平滑处理过程中可以有效保护边界信息的滤波操作方法。双边滤波能在保持图像轮廓清晰的情况下进行噪声的去除，它能自动判断滤波核处于“平坦”区域还是“边缘”区域：如果滤波核处于“平坦”区域，则会使用类似高斯滤波的算法进行滤波；如果滤波核处于“边缘”区域，则加大“边缘”像素的权重，尽可能地让这些像素值保持不变。

原图



高斯滤波效果

双边滤波效果

五 双边滤波器

OpenCV将双边滤波器封装成bilateralFilter()方法，其语法如下：

dst = cv2.bilateralFilter(src, d, sigmaColor, sigmaSpace, borderType)

参数说明：

src：被处理的图像。

d：以当前像素为中心的整个滤波区域的直径。如果d<0，则自动根据sigmaSpace参数计算得到。该值与保留的边缘信息数量成正比，与方法运行效率成反比。

sigmaColor：参与计算的颜色范围，这个值是像素颜色值与周围颜色值的最大差值，只有颜色值之差小于这个值时，周围的像素才进行滤波计算。值为255时，表示所有颜色都参与计算。

sigmaSpace：坐标空间的σ（sigma）值，该值越大，参与计算的像素数量就越多。

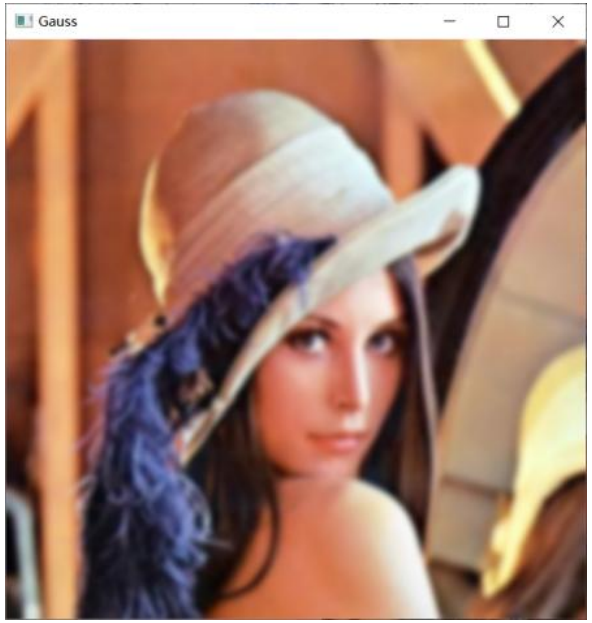
borderType：可选参数，边界样式，建议默认。

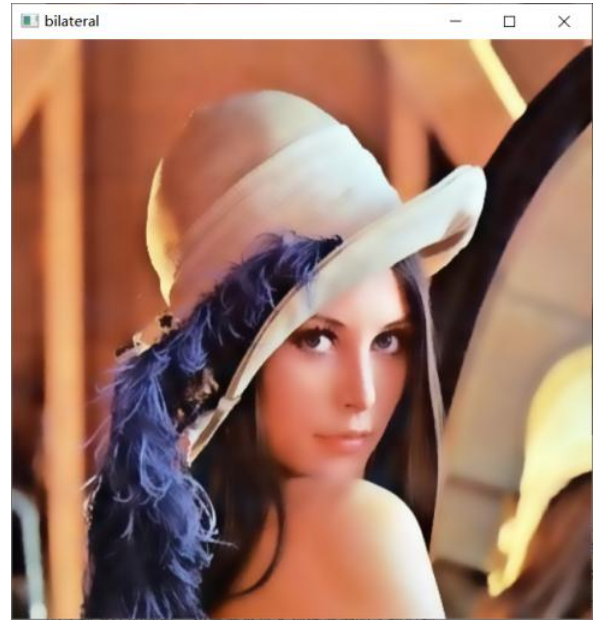
返回值说明：dst：经过双边滤波处理之后的图像。

例：对比高斯滤波和双边滤波的处理效果：使用大小为(15， 15)的滤波核对lenna图像进行高斯滤波处理，同样使用15作为范围直径对lenna图像进行双边滤波处理，观察两种滤波处理之后的图像边缘有什么差别。

```
import cv2

img = cv2.imread("lenna.jpg") # 读取原图

dst1 = cv2.GaussianBlur(img, (15, 15), 0, 0) # 使用大小为15*15的滤波核进行高斯滤波

# 双边滤波，选取范围直径为15，颜色差为120

dst2 = cv2.bilateralFilter(img, 15, 120, 100)

cv2.imshow("img", img) # 显示原图

cv2.imshow("Gauss", dst1) # 显示高斯滤波效果

cv2.imshow("bilateral", dst2) # 显示双边滤波效果

cv2.waitKey() # 按下任何键盘按键后

cv2.destroyAllWindows() # 释放所有窗体
```

例：对比高斯滤波和双边滤波的处理效果：使用大小为(15, 15)的滤波核对lenna图像进行高斯滤波处理，同样使用15作为范围直径对lenna图像进行双边滤波处理，观察两种滤波处理之后的图像边缘有什么差别。





高斯滤波模糊了整个画面，但双边滤波保留了较清晰的边缘信息。双边滤波有一个经典的应用：美颜滤镜。

小结

1.均值滤波器：中央像素取平均值，效果像马赛克。

2.中值滤波器：中央像素取排序后的中间值，效果像水彩画。

3.高斯滤波器：按照卷积核权重计算中央像素值，效果像毛玻璃效果。

4.双边滤波器：保留边缘信息，边缘清晰。

THANKS