计算机视觉

空间域图像处理（2）

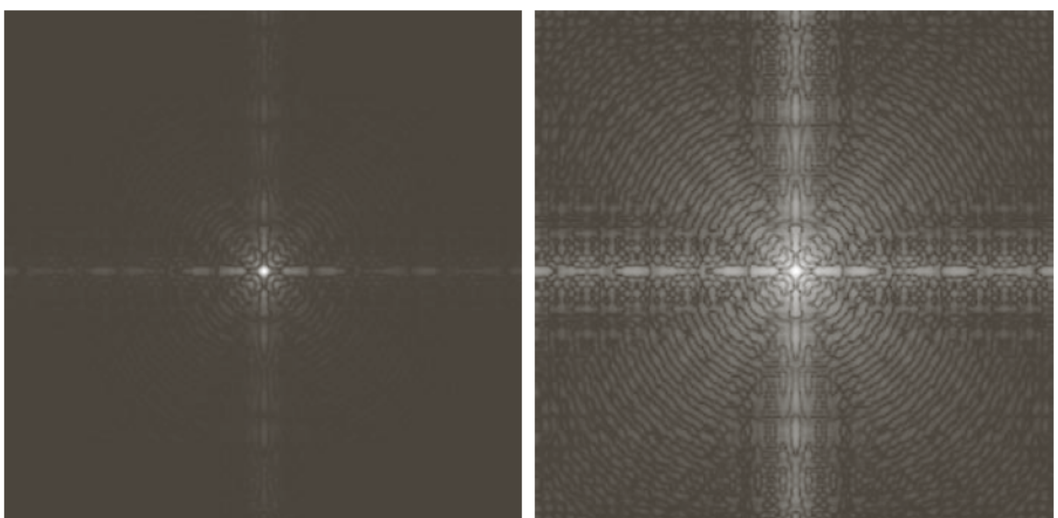
2019-3-11

上节课回顾：基于点运算的空间域处理

空间域图像处理：

$$g(x,y)=T\left[f(x,y)\right]$$

对数变换：$s=c\log(1+r)$



幂次变换：$s=cr^{\gamma}$

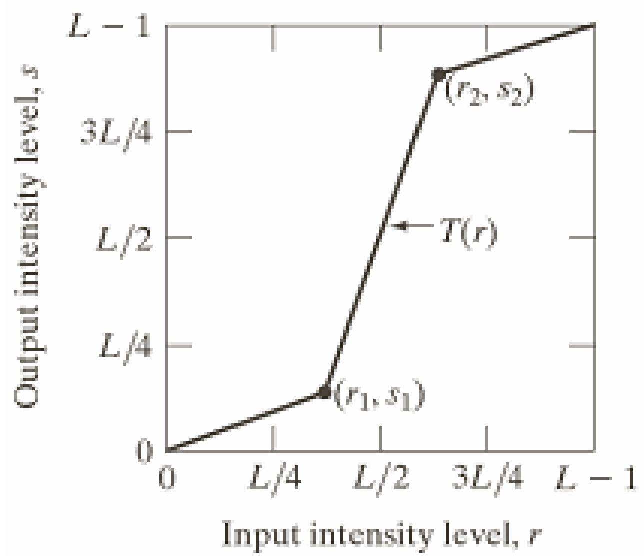
空间域图像处理

- 基本概念
- 基本运算
  - ✓ 点运算
  - ✓ 直方图运算
  - ✓代数运算
- 空间滤波器
  - ✓ 平滑空间滤波器
  - ✓ 锐化空间滤波器

点运算：对比拉伸

- 对比拉伸(Contrast Stretching): 分段线性转换方法

$$
\begin{cases}
s = \dfrac{s_1}{r_1} r, & \text{if } r < r_1 \\[2ex]
\dfrac{s - s_1}{s_2 - s_1} = \dfrac{r - r_1}{r_2 - r_1}, & \text{if } r_1 \le r \le r_2 \\[2ex]
\dfrac{s - s_2}{L - 1 - s_2} = \dfrac{r - r_2}{L - 1 - r_2}, & \text{if } r > r_2
\end{cases}
$$



其中 r:原始图像，s:转换后的图像

r_1 , r_2 , s_1 , s_2 ： 分段参数

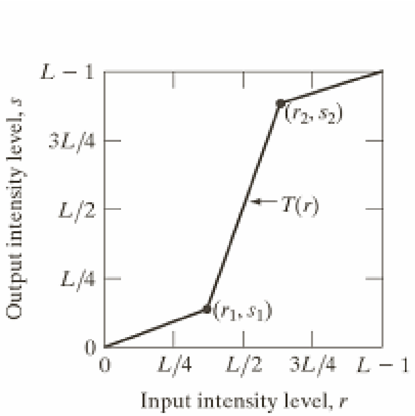
- 作用：提高对比度，在不同的灰度区间采用不同的拉伸比例。

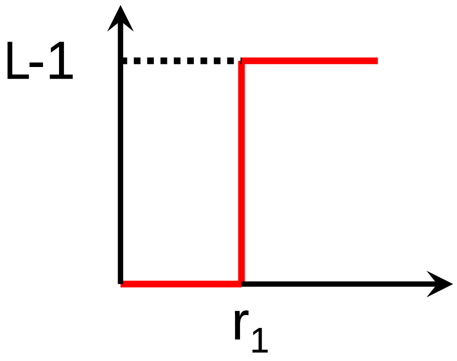
门限函数

- 门限函数(Thresholding Function)

  如果$r_1 = r_2$, $s_1 = 0$，$s_2 = L-1$，上述函数变为门限函数

$$\begin{cases} s = 0, & \text{if } r < r_1 \\ s = L-1, & \text{if } r \geq r_1 \end{cases}$$





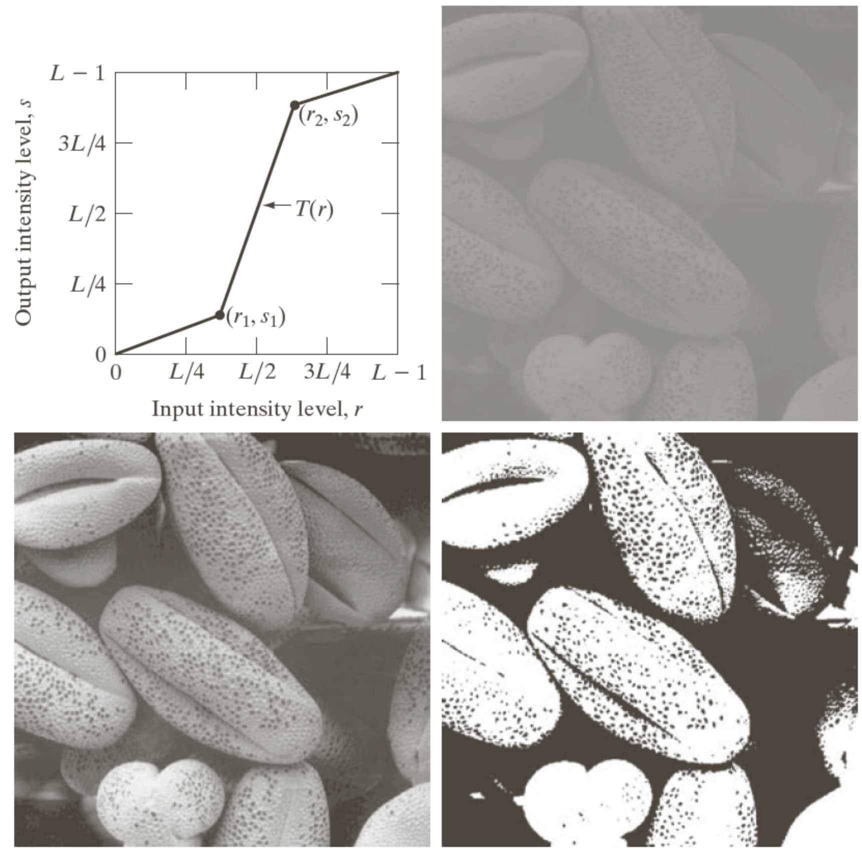
对比拉伸：示例

- 左上方：分段函数
- 右上方：原图
- 左下方：对比度拉伸的结果

  $r_1 = r_{min}$, $s_1 = 0$,

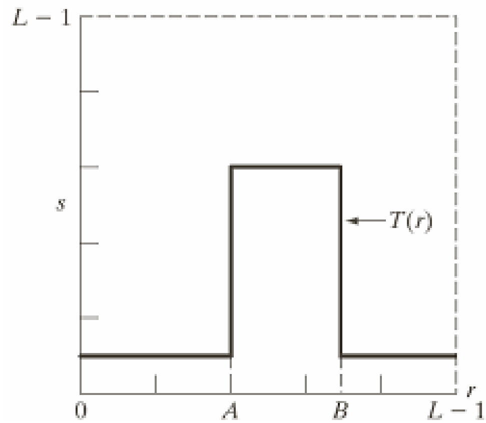
  $r_2 = r_{max}$, $s_2 = L-1$,
- 右下方：门限化的结果

  $r_1 = m$（均值）

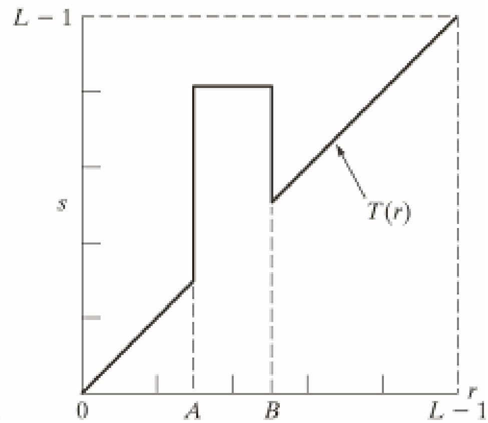


点运算：灰度切片

- 灰度切片(Gray Slicing)
- 目的：提高特定灰度范围的亮度
- 操作1：把所关心范围内的灰度指定一个较高值，而其它范围内的灰度指定一个较低值。
- 操作2：把所关心范围内的灰度指定一个较高值，而其它范围内的灰度不变。

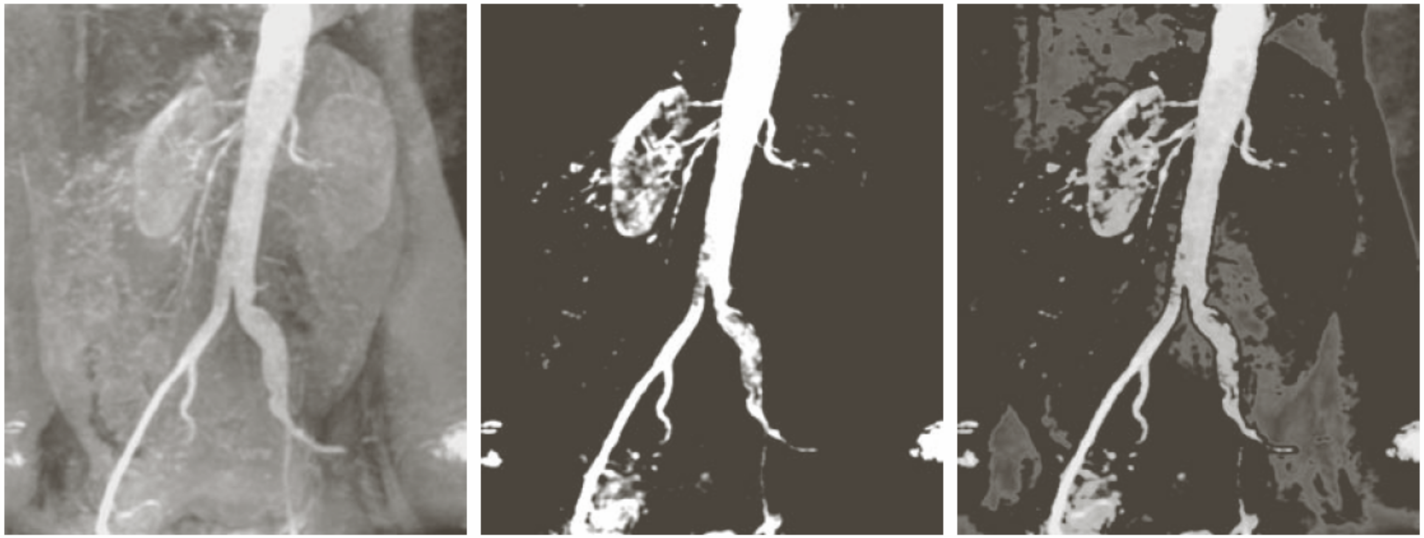


操作1



操作2

灰度切片：示例

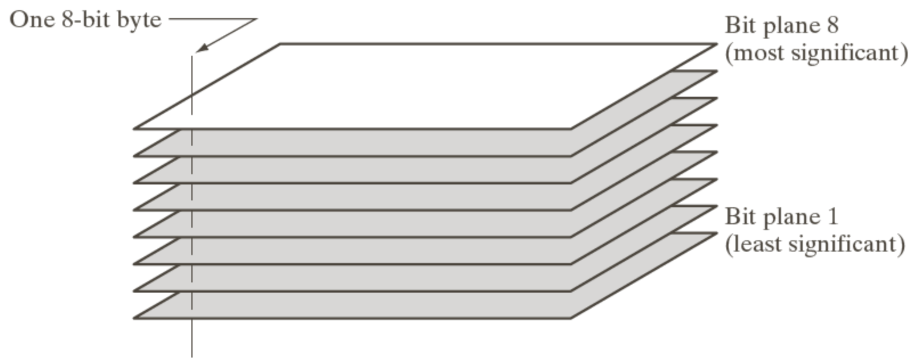


- 左：原图
- 中：灰度切片操作1
- 右：灰度切片操作2
- 问题：如何确定所关心目标的灰度范围呢？或者说，哪些是重要的需要加强的信息呢？

点运算：位平面切片

- 位平面切片(Bit-plane Slicing)

假设图像中每个像素的灰度级是256，因此可以用8位(bits)来表示每个像素的灰度。如果每一位都用一个平面(plane)来代表，位平面1代表像素中的最低位，位平面8代表像素中的最高位。因此，一副图像就可以分解为8个二进制的位平面。



点运算：位平面切片

- 较高的位包含大多数重要的视觉数据。
- 较低的位对图像的微小细节有作用。
- 可见，将图像分解为位平面，可以分析每一位在图像中的重要性。
- 因此，可以通过增强特定的位平面，达到改善图像质量的目的。

- 第一排：左：原图；中：位平面1；右：位平面2；
- 第二排：左：位平面3；中：位平面4；右：位平面5；
- 第三排：左：位平面6；中：位平面7；右：位平面8。

空间域图像处理

- 基本概念
- 基本运算
  - ✓ 点运算
  - ✓ 直方图运算
  - ✓ 代数运算
- 空间滤波器
  - ✓ 平滑空间滤波器
  - ✓ 锐化空间滤波器

直方图的定义

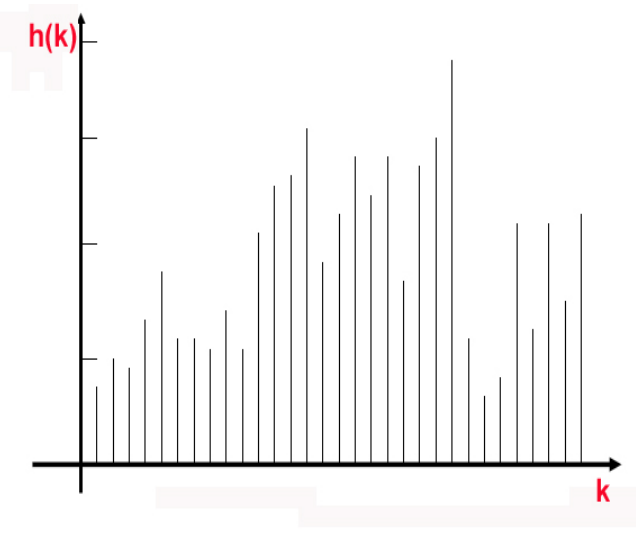
- 直方图(Histogram)：

  定义：给定一个灰度级范围[0, L-1]，一幅图像的直方图是一个离散函数：

$$h(r_k) = n_k$$

其中：r_k是灰度级，k=0, 1, …, L−1

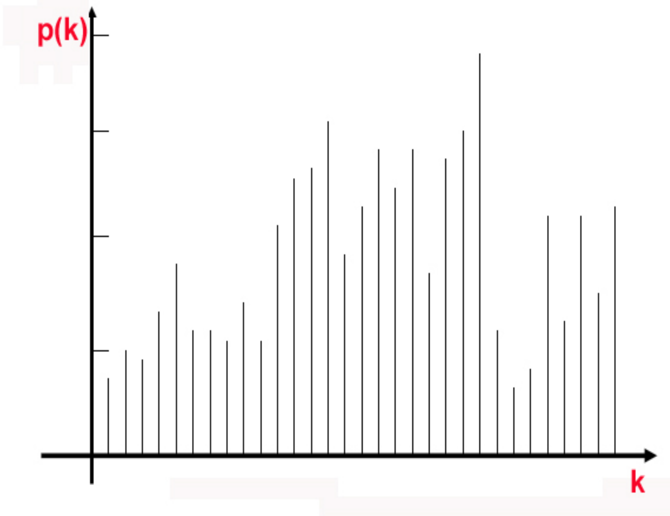
n_k是灰度级为k的像素个数。



直方图的定义：归一化

- 归一化直方图(Normalized Histogram)：使得概率之和为1.
- 归一化定义：给定一个灰度级范围[0, L-1]，一幅图像的直方图是一个离散函数：

$$p(r_k) = \frac{n_k}{n}$$



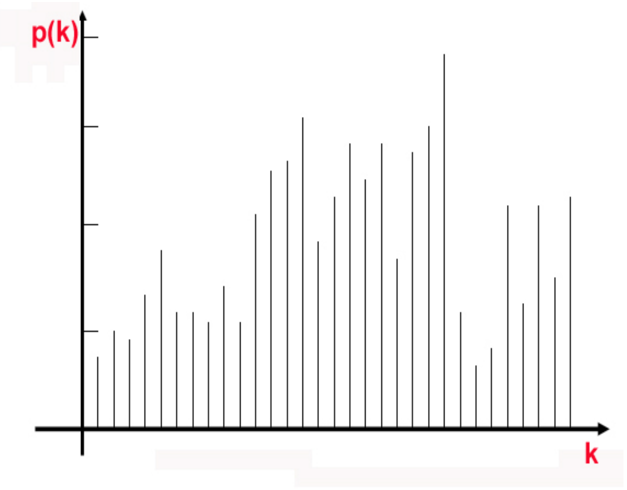
其中：r_k是灰度级，k=0, 1, …, L−1

n_k是灰度级为k的像素个数

n是图像的像素总数

归一化直方图的特性

- 归一化直方图：函数值正则化到[0,1]区间
- 归一化直方图：函数值的范围与像素总数无关。
- 归一化直方图给出了每一个灰度级在图像中出现的概率密度统计，因此它可以用来近似描述图像灰度值的概率分布。
- 直方图是多种空间域处理技术的基础，被广泛应用到图像增强、图像压缩、图像分割等领域。



直方图均衡化

- 直方图均衡化（Histogram Equalization):
  - 目的：设计一种变换，使得转换后的图像的像素占有全部可能的灰度级，且均匀分布，具有高对比度。
  - 基本思想：把原始图的直方图变换为均匀分布的形式，从而增加了像素灰度值的动态范围，达到增强图像整体对比度的效果。

直方图均衡化：连续变换函数

- 首先考虑连续函数情况，将图像灰度r归一化到[0,1]范围。

$$s = T(r) \qquad 0 \le r \le 1$$

- 转换函数 T(r) 需要满足如下两个条件：
- (1) T(r) 在区间 $0 \leqslant r \leqslant 1$ 内为单值且单调递增
- (2) 当$0 \leqslant r \leqslant 1$ 时， $0 \leqslant T(r) \leqslant 1$

- 条件(1)保证原图各灰度级在变换后仍保持从黑到白（或者从白到黑）的排列次序
- 条件(2)保证变换前后灰度值动态范围的一致性

直方图均衡化：连续变换函数

- 一种重要的变换函数：

$$s = T(r) = \int_0^r p_r(\omega)d\omega$$

其中，p_r：原图的灰度概率分布函数

- 满足条件(1)：因为p_r>0，所以p_r的积分是单值且单调递增的。
- 满足条件(2)：因为0≤r≤1， 0≤p_r≤1，所以0≤T_r≤1。

直方图均衡化：连续变换函数

- 给定p_r和$T(r)$，且$T^{-1}(s)$满足条件(1)，根据概率理论，变换后的图像灰度概率分布函数p_s则可以表达如下：

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right|$$

- 根据莱布尼茨法则（关于上限的定积分的导数是该上限的被积函数）：

$$\frac{ds}{dr} = \frac{dT(r)}{dr} = \frac{d}{dr}\left[\int_0^r p_r(\omega)d\omega\right] = p_r(r)$$

- 带入上式：

$$p_s(s) = p_r(r)\left|\frac{1}{p_r(r)}\right| = 1 \qquad 0 \le s \le 1$$

均匀分布
(Uniform Distribution)

直方图均衡化：离散变换函数

- 对于离散值（即离散概率分布）：

$$p_r(r_k) = \frac{n_k}{n}$$

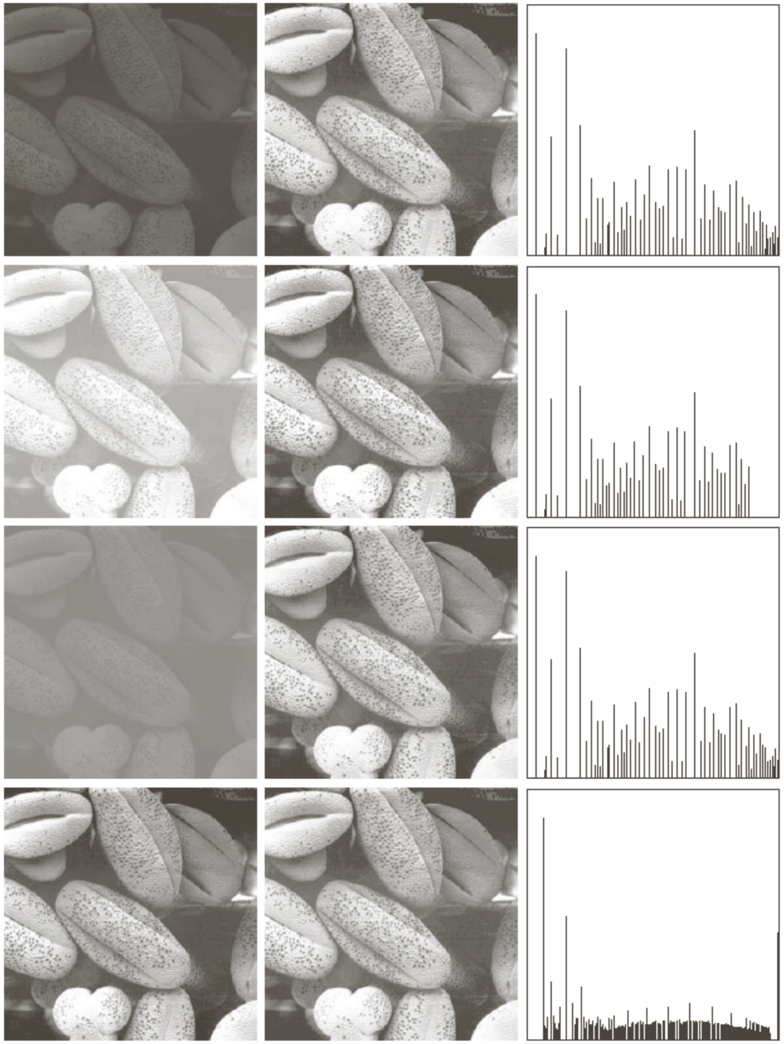
- 该变换函数的离散形式为：

$$s_k = \sum_{j=0}^{k} p_r(r_k) = \sum_{j=0}^{k} \frac{n_j}{n}$$

- 如何根据均衡化的直方图，得到输出图像？
  将输入图像中灰度级为r_k的各像素映射到输出图像中灰度级为s_k的对应像素。

直方图均衡化：示例

- 左列：原图
- 中列：直方图均衡化结果
- 右列：均衡化的直方图



空间域图像处理

- 基本概念
- 基本运算
  - ✓ 点运算
  - ✓ 直方图运算
  - ✓ 代数运算
- 空间滤波器
  - ✓ 平滑空间滤波器
  - ✓ 锐化空间滤波器

代数运算

- 代数运算：以像素对像素为基础、在两幅或多幅图像间进行。
- 算术运算：加、减、乘
- 逻辑运算：非、与、或、异或

减法运算

- 图像减法运算：两幅图像f(x,y)和h(x,y)之间的差值

$$g(x,y) = f(x,y) - h(x,y)$$

- 主要应用：
  - ✓ (1) 检测同一场景（背景）下，两幅图像之间的变化
  - ✓ (2) 图像分割：在静态背景中，获取运动物体。

减法运算：示例

- 在静态背景中，获取运动物体。

加法运算：图像均值去噪

- 图像加法运算：两幅图像f(x,y)和h(x,y)之间的和。

$$g(x,y) = f(x,y) + h(x,y)$$

- 主要应用：图像均值去噪
- 由于噪声的干扰，对于一幅原始图像f(x,y)，都会存在着一个噪声图像集合：

$$\{g_i(x,y)\}, \qquad i = 1,2,\ldots,N$$

其中

$$g_i(x,y) = f(x,y) + h_i(x,y)$$

加法运算：图像均值去噪

- 对所有噪声图像采用均值操作：

$$\overline{g}(x,y)=\frac{1}{N}\sum_{i=1}^{N}g_i(x,y)$$

- 假设噪声h(x,y)期望值为0，且互不相关，可以证明均值图像的期望值即是原始图像：

$$E[\overline{g}(x,y)]=f(x,y)$$

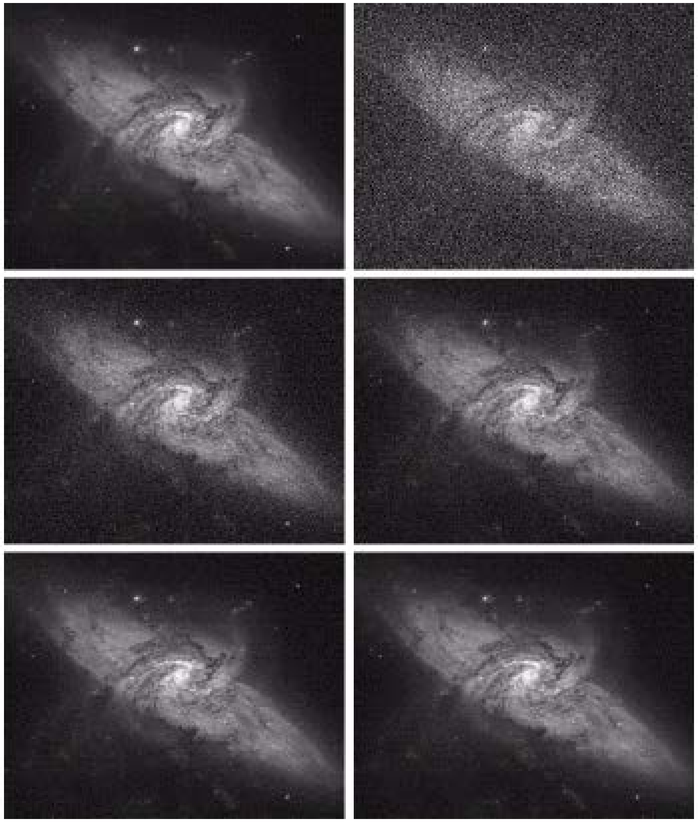
- 随着N的不断增大，均值图像会不断接近于原图像，即起到了去噪的效果。

图像均值去噪：示例

- 星系图去噪

原图

噪声图像



N=8

N=16

N=64

N=128

图像均值去噪：示例

- 星系图去噪