# 直方图的计算、傅里叶的计算、卷积的、概念简答题

# Brief Introduction of DIP

**## 1. 图像数字化过程**

图像的数字化过程分为两个步骤：采样Sampling与量化Quantization

**### 2.1 采样：** 将空间上连续的图像变换成离散点的操作称为采样。

- 采样间隔：采样点之间的距离

- 采样孔径：采样点的形状，以及是否重叠

**### 2.2 量化：** 将像素灰度转换成离散的整数值的过程

- 图像数据量：一个M\*N像素，灰度级数为G的图像所需存储空间为**M\*N\*g(bit)**，称为图像数据量。这是每个像素需要多少 bits 来存储 (比如 8-bit 图像中 $g=8$)。

 **总数据量 = (总像素个数) x (每个像素占用的bit数)**

**用8个bits来表示的1个像素灰度值的范围就是**2^8 =[0, 255]

为了表示一张grayscale (灰度) 图像，我们决定使用 **8-bit** (bit数) 来量化。

* 这个决定意味着我们总共有 **$2^8 = 256$** 个 **grayscale levels (灰度级)** (级别总数)。
* 这256个级别，我们用 **[0, 255]** 范围内的\*\*numerical values (数值)\*\* 来表示。
* 一张图像被sampling (采样) 成 $M \times N$ 个 **pixels (像素)**。
* 每一个 **pixel (像素)** 都被赋予了 [0, 255] 范围内的一个 **numerical value (数值)**，这个值就代表了它的 **grayscale (灰度)** 或亮度。

Bit (bit数) 决定了“级别总数”，value (数值) 是用来表示这些级别的数字，pixel (像素) 是存储这个value (数值) 的单元。

> 数字图像中不同灰度值的个数称为灰度级G，为2的整数次幂，即$G=2^g$​，g为量化bit数，一般来说G=256，灰度值范围[0, 255]，称为**8bit量化**

**## 2. 数字图像的表示**

数字图像通常以二维数组的形式进行表示：

- 黑白图像：对应像素值为0或1，用于表示白色或黑色

- 灰度图像：对应像素值范围[0, 255]，用于表示灰度，**0为纯黑，255为纯白**

- 彩色图像：使用RGB三个通道进行表示，每个通道对应一个二维数组，像素范围[0, 255]

## Point Operations/ Combining Images/ Histograms

**## 1. 直方图Histograms：**直方图显示了图像中特定强度值的像素出现的频率

- 仅有像素的统计信息，不会显示像素的位置分布，无法用于恢复原始数据

> 定义：

> 一个直方图H中每个条目h的高度h(i)定义为

> ​     $h(i) = card\{(u, v)| I(u,v)=i\}$

> 即灰度值等于i的所有像素点(u,v)的个数

**### 1.1 图像亮度 Image Brightness**

灰度图像的图像亮度被定义为图像中所有像素点灰度值的平均值：

​         $B(I) = \frac{1}{wh} \sum\_{v=1}{h}\sum\_{u=1}{w}I(u,v)$

由直方图可以容易判断出一幅图像的曝光是否正常

**### 1.2 图像对比度 Image Contrast**

对比度表示了区分图像中目标的难易程度：**亮度更亮，暗部更暗**，就是对比度增强

- 高对比度：直方图中有许多不同的强度值

- 低对比度：直方图中强度值较为集中

**### 1.3 动态范围 Dynamic Range**

动态范围表示图片中像素强度值的总个数

**## 2. Point Operation**

点操作是指按照某一函数对某一像素点的强度值进行处理。**一个新像素的值，只取决于它在原始图像中同一个位置上旧像素的值**。例如：

- 加减（变亮/暗）：$f(p)=p+k$

- 乘法（拉伸/缩小图像对比度范围）：$f(p)=k \times p$

- 其他实值函数：$\exp(x), \log(x), 1/x$​

- 量化

- 全局阈值

- gamma校正

```python

for v in range(1, h):

  for u in range(1, w):

    I(u, v) = f(I(u, v))

```

**上述操作并不取决于像素的位置特征，也称为Homogeneous Point Operation。同样，有基于像素位置的操作Non-Homogeneous Point Operation**。

**### 2.1 Clamping**

将像素值映射到指定范围[a,b]:

- $f(p) = a$ 如果$p<a$

- $f(p) = p$ 如果$a \le p \le b$​

- $f(p) = b$ 如果$p>b$​

**### 2.2 Invert**

图像反转：像素值反转后加最大像素值

$f\_{invert}(a) = a\_{\max} - a$ 最大值归零，其他值为正。

负片 Image Negatives：反转后的图片，有助于加强黑色背景下的白色/灰色细节

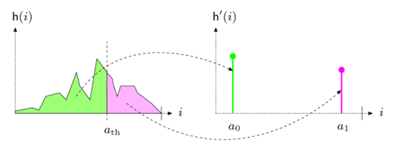
**### 2.3 Thresholding**

将超过某一阈值的像素设置为$a\_0$，小于某一阈值的像素设为$a\_1$

- $f\_{thr}(a)=a\_0$ 对所有$a < a\_{threshold}$​

- $f\_{thr}(a)=a\_1$ 对所有$a >= a\_{threshold}$​

- 当$a\_0 = 0$, $a\_1 = 1$时，该操作即为二值化，将图像转变为仅有黑白的图像



**### 2.4 基本灰度变换**

一般有三种类型，即线性变换、对数变换与指数变换

- 对数变换 Logarithmic Transformation：能够将分布于较窄范围内的像素值变换为较宽范围的分布，操作形式如下

  ​         $s = c\times \log (1+r)$

  其中r是原像素值。

- 指数变换Power Law Transformation：能够将分布于较窄范围内的较深色像素值变换为较宽范围的分布，形式如下：

  ​         $s = c \times r^\gamma$

  其中通过改变$\gamma$可以获得不同的效果

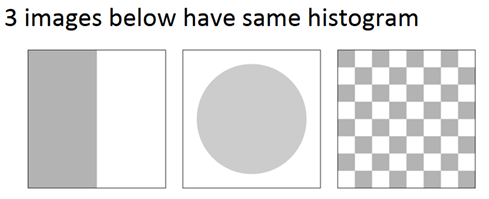
**### 2.6 Point Operation and Histograms**

有些point operation可以直观地反应在直方图中，例如改变亮度、对比度、反转等。这些操作仅对直方图的柱子进行移动或合并，并且合并操作是不可逆的。

**2. 直方图的性质 (Properties of Histograms)**

直方图告诉我们很多信息，但它也有局限性：

1. **只包含统计信息:** 直方图只告诉我们“有什么亮度”，以及“有多少”，它是一种statistical (统计) 信息 。
2. **丢失空间信息:** 直方图**完全没有**告诉你这些像素在图像中的location (位置) 。
3. **“多对一”:** 正因为丢失了空间信息，**Different images can have the same histogram (不同的图像可以有相同的直方图)** 。第10页 就展示了三个视觉上完全不同的图像，但它们的直方图是完全一样的（一半像素是灰色，一半是白色）。



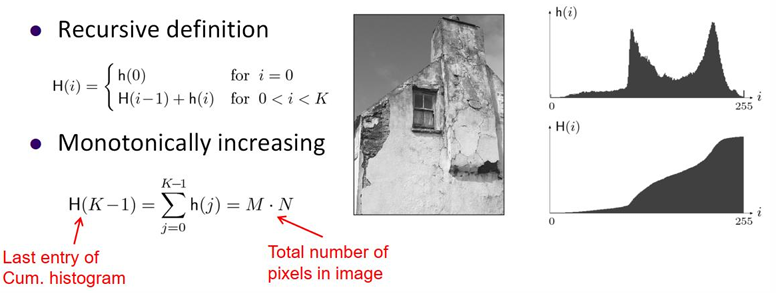
1. **不可逆:** **你无法从直方图reconstruct (重建) 出原始图像** 。

**1. 什么是累积直方图？ (What is a Cumulative Histogram?)**

Cumulative Histogram (累积直方图)，也常缩写为 **CDF (Cumulative Distribution Function，累积分布函数)**，加了一个分布之后，就变成了除以全部像素。回答了一个问题：

**“一张图像中，有多少像素的亮度值 *小于或等于* 某个级别 $i$？”** 2

它不再是统计单个亮度级别 $i$ 出现了多少次，而是统计**从0到 $i$ (包含 $i$)** 这个区间内，总共出现了多少个像素。

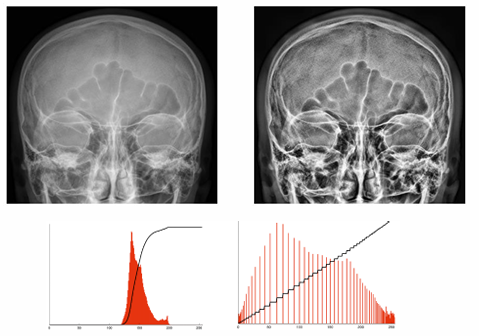


**CDF (Cumulative Distribution Function) 的主要特性 (Key Properties)**

* **Monotonically increasing (单调递增):** 累积直方图的值**永远不会下降**。它只会增加，或者在 $h(i)=0$ 的地方保持不变。
* **最后一个值是总像素数:** 累积直方图的最后一个值，即 $H(255)$（对于8-bit图像），等于图像中所有像素的总和，也就是 **$** H(255) = **M \times N$ (图像的宽 x 高) ，因为一直在累加**。

**Cumulative Histogram (累积直方图)** 是 **Histogram Equalization (直方图均衡化)** 的数学基础。

直方图均衡化（我们马上会讲到）是一种强大的enhancement (增强) 技术，它的目标就是把一个图像的直方图变得“平坦”，从而**拉伸对比度**。



**## 3. 自动对比度调整**

自动对比度调整，就是将一个范围内的像素值映射至另一个像素范围。

**## 4. Histogram Equalization**

直方图均衡化（Histogram Equalization）是一种增强图像对比度的方法，其主要思想是将一副图像的***直方图分布变成近似均匀分布***，从而增强图像的对比度。

**仅凭直方图就能判断一张照片的exposure (曝光) 质量**。

1. Underexposed (曝光不足):
   * **图像表现：** 整体太暗。
   * **直方图表现：** 大部分像素都**挤在左侧**（暗部区域），而右侧（亮部区域）几乎是空的 。
2. Overexposed (曝光过度):
   * **图像表现：** 整体太亮，亮处“白茫茫一片”，丢失了细节。
   * **直方图表现：** 大部分像素都**挤在右侧**（亮部区域）。在最右端(255)处经常有一个大“尖峰”，这叫作saturation (饱和) 。
3. Properly Exposed (曝光正常):
   * **图像表现：** 明暗分明，细节清晰。
   * **直方图表现：** 像素值\*\*spread out (分布广泛)\*\* ，很好地利用了从暗到亮的所有范围 。

**Contrast (对比度)** 也和这个概念紧密相关 。

* **Low contrast (低对比度)** 图像 (第32页 ) 的直方图会**挤在一个很窄的范围内**。
* **High contrast (高对比度)** 图像 (第32页 ) 的直方图会**分布得非常广** 。

我们已经知道了什么是***“好”的直方图（分布广）和“坏”的直方图（挤在一起）***。

那么下一步自然就是：我们能不能用一个point operation (点操作) **自动地**把一个“坏”的直方图（低对比度）**“拉伸”** (stretching) 成一个“好”的直方图（高对比度）呢？

这就是 **Contrast Stretching (对比度拉伸)** 和 **Histogram Equalization (直方图均衡化)** 要做的事情。

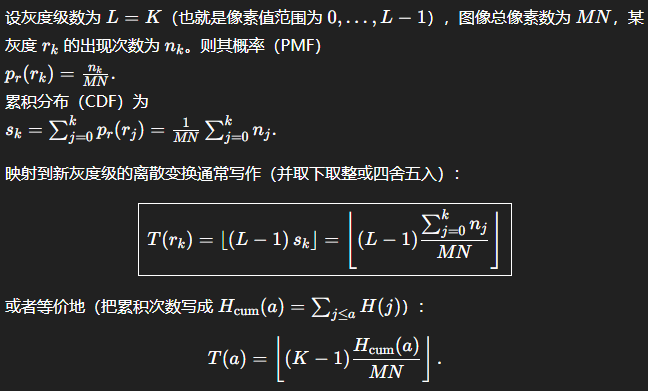
以下是**Histogram Equalization**直方图均衡化的基本步骤：

1. **\*\*计算直方图\*\***：首先，我们需要计算图像的直方图，即每个灰度级别的像素数量。

2. **\*\*计算累积分布函数（CDF）\*\***：然后，我们计算直方图的累积分布函数。这是通过对直方图的每个值进行累积求和来实现的。***这里是概率***

3. **\*\*应用映射函数\*\***：最后，***我们使用累积分布函数作为映射函数，将原始图像的灰度级别映射到新的灰度级别。这个映射函数的目标是使得输出图像的直方图尽可能地均匀分布在所有灰度级别上***。

这种方法通常用来增加许多图像的全局对比度，尤其是当图像的有用数据的对比度相当接近的时候。通过这种方法，亮度可以更好地在直方图上分布。这样就可以用于增强局部的对比度而不影响整体的对比度，直方图均衡化通过有效地扩展常用的亮度来实现这种功能。



**Contrast Stretching (对比度拉伸)**

这是最直观的增强对比度的方法。

* **目标：** 找到图像中最暗的像素值 $a\_{low}$ 和最亮的像素值 $a\_{high}$，然后用一个linear (线性) 的point operation (点操作) 将这个 [a\_{low}, a\_{high}]$ 范围“拉伸”到整个可用的范围，比如 [0, 255]`。
* 映射函数 (Mapping Function):

这个point operation (点操作) 的函数 $f(a)$ 在第36页 3 给出：

对于一个8-bit的灰度图 ( $a\_{min}=0, a\_{max}=255$ )，公式简化为 4：

* **例子：** 假设一张低对比度图像的像素值都在 [50, 150] 之间。
  + $a\_{low} = 50$, $a\_{high} = 150$。
  + 一个值为 50 的像素会变成 $(50 - 50) \times (255 / (150-50)) = 0$ (纯黑)。
  + 一个值为 150 的像素会变成 $(150 - 50) \times (255 / (150-50)) = 255$ (纯白)。
  + 一个值为 100 的像素会变成 $(100 - 50) \times (255 / (150-50)) = 127.5$ (大约 128)。

**缺点：** 这种简单的线性拉伸，虽然扩展了范围，但它并不能均匀地分布像素。现在，我们来看一个更强大、更常用的技术：**Histogram Equalization (直方图均衡化)**。

**Histogram Equalization (直方图均衡化)**

**目标**：**Histogram Equalization (直方图均衡化) 是一种non-linear (非线性) 的point operation (点操作)，它的目标是重新分布图像的像素亮度值，使得最终的直方图尽可能地\*\*flat (平坦)\*\*** 。

“平坦”意味着什么？ 意味着在 0 到 255 之间的每一个亮度级别，都有（大致）相同数量的像素。

为什么这能增强对比度？

* 在low-contrast (低对比度) 图像中，大多数像素都挤在几个亮度级别上（比如直方图中的某个大“山峰”）。
* Equalization (均衡化) 会把这个“山峰”上的像素\*\*“摊开”\*\* (spread out) 到更宽的亮度范围 。
* 原本挤在一起的、难以区分的亮度值被拉开了，图像的details (细节) 就显现出来了。

**3. Histogram Equalization (直方图均衡化) 的神奇之处：如何实现？**

我们如何找到那个能把直方图“拉平”的mapping function (映射函数) $f(a)$ 呢？

**答案就是：这个神奇的映射函数 $f(a)$ 正是图像的Cumulative Distribution Function (CDF) (累积分布函数)！**

我们在前面刚学过，CDF (累积分布函数) 就是归一化（normalized）的\*\*Cumulative Histogram (累积直方图)\*\*。

实现Histogram Equalization (直方图均衡化) 的四步pseudo-code (伪代码)：

1. **计算直方图 (Calculate Histogram):**

遍历图像，统计每个亮度值 $i$ (0-255) 出现了多少次，得到 $h(i)$。

1. **计算累积直方图 (Calculate Cumulative Histogram / CDF):**

计算 $H(i)$。

* + $H(0) = h(0)$
  + $H(i) = H(i-1) + h(i)$ 如此累加，导致H(255)=all+

1. 标准化并缩放 (Normalize and Scale the CDF):

这是最关键的一步：创建lookup table (查找表) / mapping function (映射函数) $f(a)$。

对于每一个亮度值 $a$ (从 0 到 255)，它的新值 $f(a)$ 由以下公式决定：

* + $H(a)$：累积直方图在 $a$ 处的值。
  + $total\\_pixels$：*总像素数 (即 $H(255)$)*。
  + $(L-1)$：最大亮度值 (对于8-bit图像，就是 255)。
  + $\frac{H(a)}{total\\_pixels}$ 这一步就是normalization (归一化)，把累积计数值转换成 $[0, 1]$ 范围内的概率。

1. 变换图像 (Transform the Image):

创建一张新图像。遍历原始图像的每一个像素 $(u,v)$，读取它的旧值 $a = I(u,v)$，然后在新图像的相同位置写入新值 $a^{\prime} = f(a)$。

第43页的图示非常清晰地展示了结果：

* **左上角：** 原始的低对比度图像，其直方图 $h(i)$ 挤在左侧（暗部）。
* **右侧：** “Transformation function (变换函数)”——这就是我们刚刚计算出的\*\*CDF (累积分布函数)\*\*。
* **左下角：** 经过Equalization (均衡化) 后的新图像，对比度大大增强，其直方图 $h'(i)$ 变得非常spread out (分散)。

总结一下：

* **Contrast Stretching (对比度拉伸)**：**线性**变换，只关心 $a\_{min}$ 和 $a\_{max}$，会产生gaps (间隙)。
* **Histogram Equalization (直方图均衡化)**：**非线性**变换，使用CDF (累积分布函数) 作为映射函数，试图创造一个flat (平坦) 的直方图，对比度增强效果更强。

**## 6. Histogram Matching**

**1. 什么是 Histogram Matching (直方图匹配)？**

* **Histogram Equalization (直方图均衡化)** 的目标是把histogram (直方图) 变得 **flat (平坦)**。
* **Histogram Matching (直方图匹配)** 更进了一步：它的**目标是任意指定一个target histogram (目标直方图)，然后通过point operation (点操作) 把原始图像的histogram (直方图) 变成那个你想要的形状**。

**2. 它有什么用？(Why use it?)**

Histogram Matching (直方图匹配) 最酷的应用就是 **Appearance Transfer (外观迁移)**。

正如你在 lecture-3-hist.pdf 的第62页和第63页所看到的：

* 你有一张“Original (原始)”图像（比如一张阴天的城市照片）。
* 你还有一张“Reference (参考)”图像（比如一张色彩鲜艳的日落照片）。
* Histogram Matching (直方图匹配) 可以修改“Original (原始)”图像，让它的histogram (直方图)（颜色和亮度分布）去匹配“Reference (参考)”图像的histogram (直方图)。
* **结果 (Result)**：你的阴天照片就会被赋予日落照片的contrast (对比度) 和color style (色彩风格)。

**3. 它是如何工作的？(The Algorithm)**

这个算法非常巧妙，它**同时利用**了原始图像和目标图像的 **Cumulative Distribution Function (CDF) (累积分布函数)**。

我们把Histogram Equalization (直方图均衡化) 想象成把图像“压平”到flat (平坦) 的中间状态，Histogram Matching (直方图匹配) 则是**先“压平”再“塑形”**。

该方法的思想是：找到目标像素概率中，不小于原始图像像素概率所对应的最小的像素值

**## 7. Gamma校正**

回顾基本的成像与显示原理，光线经过采集设备（相机等）转变为电信号进行存储，同时电信号输入显示设备进行还原。然而不同的采集设备对光线的反应是不一致的，从而导致产生的电信号也是不同的。这里就产生了两个一致性问题：1. 如何保证在相同条件下不同设备采集到的图片是一致的；2. 如何保证同一图像在不同显示设备上的一致。

问题的关键在于以下两个映射：

- 采集设备传感器接收到的光信号映射至对应像素点的像素强度值

- 像素强度值映射至显示设备上该像素点对应的光

这两个映射都是非线性且复杂的，因此目标是找到一个可用的近似。

**### 7.1 什么是****Gamma**

Gamma值源自于模拟摄影，是基于如下Exposure Function定义的：

Exposure Function被定义为光强度的对数（B）与film density（D）的对应关系

Gamma则被定义为该曲线线性范围内的斜率，$\gamma \triangleq \frac{\Delta D}{\Delta B}$

通过这一方式，可以很好地拟合Exposure Curve。实践中采用Gamma函数的形式：$b=f\_{\gamma}(a)=a^\gamma$，其反函数为$a=f\_{\gamma}^{-1}(b) = b^{1/\gamma}$。

**### 7.2 Gamma校正**

在光线（对数强度B）进入相机后，经过gamma函数$\gamma\_c$拟合实际exposure curve，输出电信号$s=B^{\gamma\_c}$。Gamma校正是指**\*\*通过应用反伽马函数$f\_{gc}(s, 1/\gamma\_c)$获得与原始光强度 B几乎相同的测量值 b\*\***。

# # III. Filters

**## 1. Filter**

**传统的point operation对图像的操作能力有限，例如对图像的模糊、平滑以及Sharpening锐化操作**等，都无法通过该手段实现。

而Filter则通过**结合某一像素点及其邻近像素点**的像素值，能够实现更为多样的图像变换操作。

**## 2. 图像增强方法**

常规的图像增强方法包括两类：

- **空间域**图像增强技术：直接操作图像上某一位置的像素点像素值

- **频率域**图像增强技术：图像的傅立叶变换或小波变换的处理

**### 2.1 Spatial Filter**

空间滤波器，被定义为**结合某一像素点及其周围像素点的像素值**进行的操作方法。

**1. 为什么需要滤波器？(Why Filters?)**

我们在上一个主题学了 **Point Operations (点操作)**。它的特点是“一个萝卜一个坑”，新像素值**只**取决于对应的旧像素值。

但是，Point Operations (点操作) 有很多局限性。比如：

* 它无法实现 **Blurring (模糊)** 或 **Smoothing (平滑)** (如 lecture-4-filter.pdf 第14页所示 1)。
* 它无法实现 **Sharpening (锐化)** (如 lecture-4-filter.pdf 第3页所示 2)。
* 它无法有效地 **remove noise (去除噪声)** (如 lecture-4-filter.pdf 第67页所示)。

这些操作的共同点是，它们都需要\*\*combine a pixel's value with the values of its neighbors (将一个像素的值与其邻域像素的值结合起来)\*\* 4444。

这就是 **Spatial Filter (空间滤波器)** 的用武之地。

**2. 什么是 Spatial Filter (空间滤波器)？ 卷积**

Spatial Filter (空间滤波器) 的核心思想是：

一个新像素的值，是通过一个 weighted average (加权平均) 计算出来的，这个平均值的范围包括它在原始图像中的 neighborhood (邻域) 5。

这个“加权平均”的权重被定义在一个小矩阵中，这个矩阵被称为：

* **Filter Matrix (滤波器矩阵)**
* **Kernel (核)**
* **Mask (掩码)**

**Linear Filter (线性滤波器)** 的一般过程在 lecture-4-filter.pdf 的第10页 6和第19页 7 有很好的图示：

1. **对齐：** 将 filter kernel (滤波器核) 的中心（也叫 hot spot 8）对准你想要计算的原始图像中的某个像素 $(u,v)$。
2. **相乘：** kernel (核) 中的每一个权重值，都与它所覆盖的图像像素值相乘。
3. **求和：** 将所有相乘的结果加起来，得到的值就是新图像中 $(u,v)$ 位置的像素值。
4. **滑动：** 对图像中的每一个像素重复这个过程。

**3. Correlation (相关) vs. Convolution (卷积)**

Correlation (相关) 和Convolution (卷积) 是实现滤波的两种非常相似的数学运算。在你的课程中，它们经常被（不严格地）混用，但它们的数学定义有细微但重要的差别。

* **Correlation (相关):**
  + **过程：** 正如我们上面所描述的。Filter kernel (滤波器核) **直接**在图像上滑动，进行“乘-加”操作。
* **Convolution (卷积):** **在整幅图像上应用滤波器的操作被称为卷积Convolution。**
  + **过程：** 在执行“乘-加”操作**之前**，filter kernel (滤波器核) $H$ 必须先\*\*flip 180 degrees (旋转180度)**（*即水平翻转并且垂直翻转***）。
  + 公式 (如 lecture-4-filter.pdf 第12页所示 11)：
  + 注意公式中是 $I(u-i, v-j)$，这等效于将 $H$ 翻转。

关键点：在图像处理中，我们使用的大部分 kernels (核)（比如 Box filter (盒式滤波器) 12, Gaussian filter (高斯滤波器)）都是 **symmetric (对称)** 的。

**当一个 kernel (核) 是对称的时，旋转180度后它还是它自己。在这种情况下，Correlation (相关) 和 Convolution (卷积) 的计算结果是完全一样的** 14。

**最基本的linear filters (线性滤波器)，从 Blurring (模糊) 开始**

## 1. Smoothing Filters (平滑滤波器)

Smoothing (平滑)，也常被称为 **Blurring (模糊)**，它的主要目的是**抑制图像中的高频成分**。

* **主要用途：**
  1. **Noise Reduction (噪声抑制)：** 通过对邻域取平均，可以减弱random noise (随机噪声) 的影响。
  2. **Blurring (模糊)：** 作为一种效果，或者作为Image Pyramids (图像金字塔) （我们后面会讲）的预处理步骤。
* **核心思想：** Smoothing (平滑) 滤波器会用一个neighborhood (邻域) 内像素的\*\*weighted average (加权平均)\*\* 来替换中心像素的值，进而实现**抑制图像中的高频成分**。

**A.** **Box Filter (盒式滤波器) / Mean Filter (均值滤波器)**

这是最简单的smoothing filter (平滑滤波器)。

* **Kernel (核):** 它的kernel (核) 中所有coefficients (系数) 都相等。
* 示例 (3x3 Box Filter):

如 lecture-4-filter.pdf 第19页所示 1，这个操作就是简单地将一个 $3 \times 3$ 邻域内的9个像素值相加，然后除以9，用这个average (平均值) 替换中心像素的值。

* Filter Size (滤波器尺寸) 的影响:

Kernel (核) 越大，blurring (模糊) 效果越强。lecture-4-filter.pdf 的第27页 2 就展示了从 $7 \times 7$ 到 $41 \times 41$ 的kernel (核) 带来的逐渐增强的模糊效果。

**B.** **Gaussian Filter (高斯滤波器)**

Box filter (盒式滤波器) 有点太“硬”了，它同等地对待邻域中的所有像素。一种更平滑、更常用的smoothing filter (平滑滤波器) 是 **Gaussian Filter (高斯滤波器)**。

* **Kernel (核):** 它的权重不是平均的，而是根据一个\*\*Gaussian (高斯) 函数\*\*（即“钟形曲线”或正态分布）来分布的。
  + 位于kernel (核) **中心**的像素获得**最高的权重**。
  + 离中心**越远**的像素，获得的**权重越低**。
* 示例 (一个近似的 3x3 Gaussian Kernel):

(注意：这是一个weighted average (加权平均)，如 lecture-4-filter.pdf 第31页所示 4)

* **优势：** Gaussian (高斯) 模糊是一种更“自然”的模糊，它能产生更平滑的边缘过渡，并且在数学上有很多优良特性。

**2. Separability (可分离性)**

这是一个非常重要的计算优化概念，在 lecture-4-filter.pdf 的第52-58页 5 有详细讨论。

* **问题：** 对于一个 $M \times M$ 的kernel (核)，在 $w \times h$ 的图像上进行convolution (卷积)，计算复杂度是 $O(M^2 \times w \times h)$。当 $M$ 很大时（比如 $M=41$），$M^2$ ( $41^2=1681$ ) 会让计算变得非常慢。
* **Separable Kernel (可分离核):** 如果一个2Dkernel (核) $H$ 可以被分解 (separate) 为一个vertical (垂直) 1D kernel (核) $H\_y$ 和一个horizontal (水平) 1D kernel (核) $H\_x$ 的convolution (卷积)，即 $H = H\_y \* H\_x$，那么它就是可分离的。
* 示例 (Box Filter):
* **优势：**
  1. 先用 $H\_x$ (大小 $1 \times M$) 扫一遍图像（$M \times w \times h$ 次运算）。
  2. 再用 $H\_y$ (大小 $M \times 1$) 扫一遍结果（$M \times w \times h$ 次运算）。
  3. 总计算复杂度降为 $O(2M \times w \times h)$。
  4. **计算量从 $O(M^2)$ 降到了 $O(M)$！**
* **Gaussian Filter (高斯滤波器) 是可分离的 (Separable):** lecture-4-filter.pdf 的第57页 8 证明了2D高斯函数可以被分解为两个1D高斯函数的乘积，因此Gaussian Filter (高斯滤波器) 总是可以被高效地计算。

**3. Boundary Handling (边界处理)**

当你slide (滑动) 滤波器kernel (核) 到图像的**边缘**时，kernel (核) 的一部分会悬在图像之外 9。这时怎么办？lecture-4-filter.pdf 的第36-40页 10 展示了几种策略：

1. **Crop (裁剪):** 忽略这些像素。output image (输出图像) 会比input image (输入图像) 小一圈。
2. **Pad (填充):** 在原图周围加一圈“垫子”。
   * Zero-padding (零填充)：用 0 来填充 (最常见) 12。
   * Extend / Replicate (扩展/复制)：重复使用边缘的像素值。
   * Wrap / Periodic (环绕)：把图像想象成是周期性的，用对面的像素来填充。
   * Mirror (镜像)：像照镜子一样反射边缘的像素 15。

我们已经讲完了Smoothing Filters (平滑滤波器)。与“模糊”相对立的概念就是“锐化”。

接下来，我们要学习 **Sharpening Filters (锐化滤波器)**，比如 Laplacian (拉普拉斯) 滤波器和 Unsharp Masking (USM, 锐化掩模)。

## Sharpening Filters (锐化滤波器)

你已经了解了 Smoothing Filters (平滑滤波器)，它们通过average (平均) 来blur (模糊) 图像。现在我们来看它的对立面：**Sharpening Filters (锐化滤波器)**。

**1. 什么是锐化？ (What is Sharpening?)**

Sharpening (锐化) 的目标是\*\*highlight (突显) 图像中的 transitions (过渡区域)\*\*，也就是edges (边缘)。

* **Smoothing (平滑)** 是通过**summation (求和/平均)** 来 **blur (模糊)** 过渡。
* **Sharpening (锐化)** 是通过**differentiation (微分/求差)** 来 **highlight (突显)** 过渡。

lecture-4-filter.pdf 第44页 3的 Mach Band Effect (马赫带效应) 告诉我们，人眼在边缘处会自动“脑补”出更亮和更暗的线条（overshoots (过冲) 和 undershoots (下冲)）。Sharpening (锐化) 就是在数字上模拟并\*\*amplify (放大)\*\* 这种效果 4。

lecture-4-filter.pdf 中提到了两种主要的锐化方法：**Laplacian filter (拉普拉斯滤波器)** 和 **Unsharp Masking (USM, 锐化掩模)**。我们先重点讲Unsharp Masking (USM)。

**2. Unsharp Masking (USM)**

这是 Photoshop 等软件中最常用的锐化技术。lecture-4-filter.pdf 的第58页和第59页 6 完美地展示了它的流程：**USM (Unsharp Masking) 的三个步骤：**

1. Blur (模糊) 图像: 首先，复制一张原始图像 $I$，并对其应用一个low pass filter (低通滤波器)（比如 Gaussian filter (高斯滤波器)）来得到一张模糊图像 $\tilde{I}$。
2. 创建“掩模” (Create the Mask): 从原始图像 $I$ 中\*\*Subtract (减去)\*\* 这张模糊图像 $\tilde{I}$。

这个 $M$ 就是mask (掩模)。它包含了什么？它只包含原始图像中被模糊掉的“高频”细节，也就是edges (边缘)。

1. Add (添加) 掩模:

最后，将这个mask (掩模) $M$（通常会乘以一个weight (权重) $a$）加回到原始图像 $I$ 上。

或者，写成一个完整的公式：

* **Highboost Filtering (高增益滤波):** 当这个权重 $a$（在 lecture-5-edge.pdf 第48页 10中被称为 $k$）**大于1**时，这个操作就被称为 Highboost Filtering (高增益滤波)，它会产生非常强烈的锐化效果 11。
* **优势:** USM (锐化掩模) 比其他方法（如 Laplacian (拉普拉斯)）更好，因为它在第一步smoothing (平滑) 时就**抑制了噪声**，并且你可以通过调整blur (模糊) 的程度 $\sigma$ 和权重 $a$ 来精确控制锐化效果 12。

**3. Difference Filters (差分滤波器) - Laplacian (拉普拉斯)**

lecture-4-filter.pdf 第41页和第45页提到了另一类linear filters (线性滤波器)：**Difference filters (差分滤波器)**。

* **特点:** ***它们的kernel (核) 系数有正有负（而smoothing filters (平滑滤波器) 只有正系数）***。
* **Laplacian (拉普拉斯) 滤波器:** 这是一个difference filter (差分滤波器)，它通过计算second derivative (二阶导数) 来检测边缘，非常适合锐化 14。

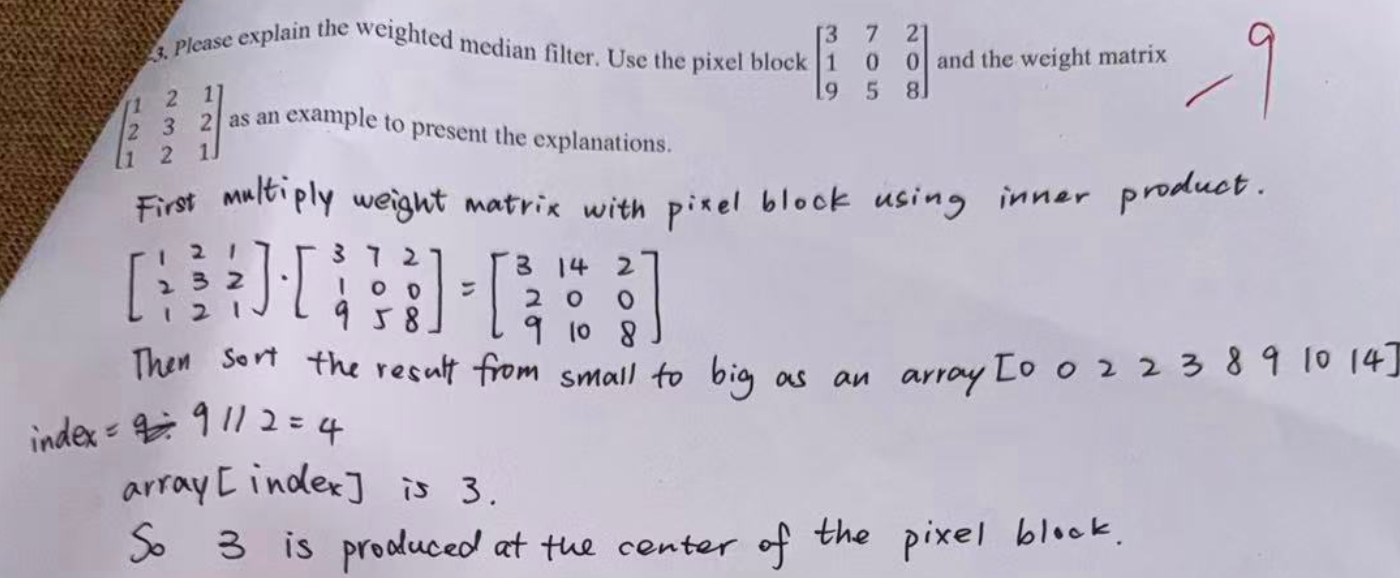
我们已经讲完了 lecture-4-filter.pdf 的核心内容：Spatial Filters (空间滤波器)，包括Correlation (相关) / Convolution (卷积) 的定义、Smoothing (平滑) 滤波器（Box, Gaussian）和 Sharpening (锐化) 滤波器（USM）。

Laplacian (拉普拉斯) 滤波器引出了一个全新的、极为重要的主题，也就是你的考纲第6-7周的内容：**Edges and contours (边缘与轮廓)**。

**Non-Linear Filters (非线性滤波器)：** Non-linear filters (非线性滤波器) 指的是那些output (输出) **像素值不是通过对input (输入) 邻域像素值进行weighted sum (加权求和)** 得到的滤波器 。它们基于non-linear (非线性) 的函数来组合像素。**Weighted Median Filter (加权中值滤波器)也是非线性**，因为还是有取中指的操作。

**中值滤波器是对框内的排序后取中值**

**加权是指，先重复权重次，然后再取中值**



## Edges and contours (边缘与轮廓)

**1. 什么是 Edge (边缘)？**

Edge (边缘) 是图像处理中最重要的feature (特征) 之一, 定义为：

**Edge (边缘) = 图像亮度 (brightness) 发生sharp change (剧烈变化) 的地方。**

这些剧烈变化（discontinuities (不连续性)）可以由多种物理因素引起：

* Depth discontinuity (深度不连续)： 一个物体挡住了另一个更远的物体。
* Surface normal discontinuity (表面法线不连续)： 一个物体的表面发生了弯折（比如盒子的拐角）。
* Surface color discontinuity (表面颜色不连续)： 物体表面的纹理或颜色发生了变化。
* Illumination discontinuity (光照不连续)： shadow (阴影) 投射到了物体表面。

**2. 如何用数学检测 Edge (边缘)？**

这个“剧烈变化”的概念，在微积分中是用 **Derivative (导数)** 来描述的。

我们先从1D的情况来看：

* **理想边缘 (step edge)：** 亮度值从低瞬间跳到高。
* **First derivative (一阶导数)：** 在这个“跳变”的地方，一阶导数（即变化率）会达到一个\*\*peak (峰值)\*\*。
* **Second derivative (二阶导数)：** 在一阶导数达到峰值的地方，二阶导数会穿过零点，这被称为 **zero-crossing (零交叉)**。

**因此，检测边缘的两种主要方法就是：**

1. **First-order (一阶) 方法：** 寻找一阶导数的peak (峰值)。
2. **Second-order (二阶) 方法：** 寻找二阶导数的zero-crossing (零交叉)。

***Derivative (导数) 滤波器（包括一阶和二阶）的系数和通常为0，这意味着它们对flat (平坦) 区域（恒定亮度）的响应为0。***

**3. 如何在 Digital Image (数字图像) 上计算导数？**

在digital image (数字图像) 中，我们没有连续函数，只有discrete (离散) 的像素网格。我们使用 **Finite Differences (有限差分)** 来近似导数。

最常用的近似方法是 **Central Difference (中心差分视为导数)** ：

* **1D 导数：**

为了归一化，常写作 。

* **用Convolution (卷积) 实现：** 这种差分操作可以通过convolution (卷积)（或correlation (相关)）一个kernel (核) 来实现。例如，计算 $x$ 方向的导数，我们可以用 **kernel 来卷积（Laplacian (拉普拉斯) 滤波器）**。

**4. 扩展到 2D 图像：Gradient (梯度)**

在2D图像中，我们使用 **Gradient (梯度)** $\nabla f$ 来表示导数。

* Gradient (梯度) 是一个\*\*vector (向量)\*\*，它有两个分量：
  + ：水平方向的导数（用 计算）。
  + ：垂直方向的导数（用 计算）。
* Gradient (梯度) 告诉我们两件事：
  1. **Direction (方向)：** 梯度向量指向intensity (强度) **增长最快**的方向，这个方向\*\*perpendicular (垂直) 于 edge (边缘)\*\* 13。
  2. **Magnitude (幅度)：** 。这个幅度代表了\*\*edge strength (边缘强度)\*\*。幅度越大，边缘越sharp (剧烈)。

**5. First-order Edge Detectors (一阶边缘检测器)**

**Prewitt 和 Sobel 算子 (Operators)**：

* **问题：** 直接在有noise (噪声) 的图像上计算导数会得到非常糟糕的结果（噪声会被放大）15。
* Derivative Theorem (导数定理)： “先对图像smooth (平滑) 再求导” 与 “先对kernel (核) 求导再convolve (卷积)” 是一样的。
* **Prewitt 算子 (Operator):** 它将\*\*averaging (均值) 滤波器\*\*（Box filter (盒式滤波器)）和\*\*derivative (导数) 滤波器\*\*（central difference (中心差分)）结合了起来。
* **Sobel 算子 (Operator):** 它和Prewitt (普鲁伊特) 类似，但它在smoothing (平滑) 时使用了\*\*weighted (加权) 平均\*\*（更像Gaussian (高斯)），给予了中心像素更高的权重。这使得Sobel (索贝尔) 算子对噪声更robust (鲁棒)，效果通常也更好。

**6.** **"最优" 检测器：*****Canny Edge Detector (Canny边缘检测)***

Canny (坎尼) 边缘检测（lecture-5-edge.pdf 第67-75页 19）是目前最常用和最有效的边缘检测器之一。它试图满足三个标准：

1. **Good Detection (良好检测)：** 不漏掉真边缘，不误报假边缘。
2. **Good Localization (良好定位)：** 检测到的边缘要尽可能接近真实边缘。
3. **Minimal Response (最小响应)：** 每条边缘只标记一次（即边缘要细），NMS。

它通过一个多步骤pipeline (流水线) 实现：

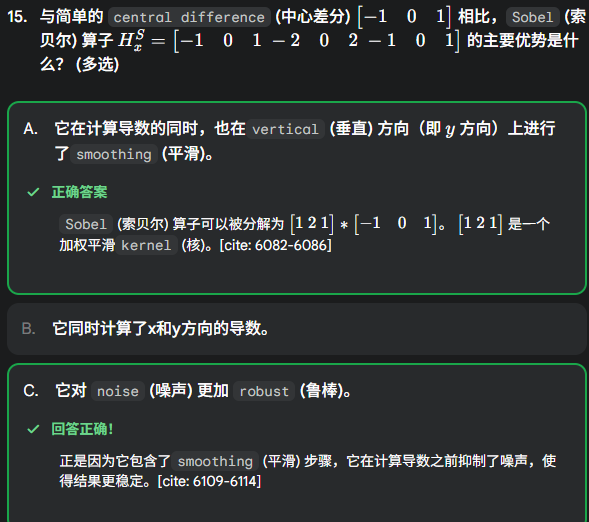
1. **Suppress Noise (噪声抑制)：** 使用Gaussian filter (高斯滤波器) 对图像进行平滑。
2. **Compute Gradient (计算梯度)：** 使用Sobel (索贝尔) 算子计算gradient magnitude (梯度幅值) 和direction (方向) 22。
3. **Non-Maxima Suppression (非极大值抑制)：**
   * **目的：** 将gradient magnitude (梯度幅值) 图中检测到的“粗”边缘变thin (细)。
   * **方法：** 沿着gradient direction (梯度方向) 23，检查每个像素是否是其邻域中的local maxima (局部最大值)。如果它不是“山峰”的“山脊”，就将其suppress (抑制) 掉（设为0）。通过只保留梯度方向上的局部最大值，NMS 可以将由梯度产生的**宽“山脊”细化为一条单像素宽的线**。
4. **Hysteresis Thresholding (双阈值/滞后阈值)：**
   * **目的：** 消除由noise (噪声) 引起的spurious (假的) 边缘，同时connect (连接) 起true (真的) 边缘。
   * **方法：** 使用**两个**阈值，$T\_{low}$ 和 $T\_{high}$ 25。
     + $ > T\_{high}$ 的像素 = **strong edge (强边缘)** (一定是边缘)。
     + $ < T\_{low}$ 的像素 = **non-edge (非边缘)** (立即丢弃)。
     + 在 $T\_{low}$ 和 $T\_{high}$ 之间的像素 = **weak edge (弱边缘)**。
   * **Connectivity Analysis (连通性分析)：** 只有当一个weak edge (弱边缘) 像素连接到了一个strong edge (强边缘) 像素时，它才会被保留。这个过程就像从strong edge (强边缘) 开始“生长”，连接所有相邻的weak edge (弱边缘) 。

lecture-5-edge.pdf 第74页清楚地展示了这几个步骤（梯度 -> NMS -> Hysteresis）是如何逐步提炼出最终边缘的。

我们已经深入探讨了Edges (边缘)。考纲的下一部分是 **Corner detection (角点检测)**。这在 lecture-7-corner-line.pdf 中有详细介绍。







***这里错在没有分解，分解之后不是全正数，所以不算平滑算子，不鲁棒。***

***是否抗噪的关键 = 该算子能否分解出低通滤波（平滑）部分。***

| **算子类型** | **数学原理** | **空间滤波器结构特征** |
| --- | --- | --- |
| **一阶导算子** | 寻找**变化率的峰值（Peak）**。 | **核系数通常从负值平滑过渡到正值** 2。通常用于计算梯度（Gradient）。 |
| **二阶导算子** | 寻找**零交叉（Zero-crossing）**（即一阶导数达到峰值的地方）。 | **核中心系数符号与周围系数符号相反**，反映 $x\_{i+1} - 2x\_i + x\_{i-1}$ 的结构。 |

***所谓平滑***：用一个**neighborhood** (邻域) 内像素的\*\***weighted average (加权平均)**\*\* 来替换中心像素的值，进而实现**抑制图像中的高频成分**。

***low-pass filter (低通滤波器)对应低频，对应模糊化处理；***

***高通滤波器对应高频（边缘，噪声都是高频），对应锐化处理***

| **特性 (Property)** | **Low-Pass Filters (低通滤波器)** | **High-Pass Filters (高通滤波器)** |
| --- | --- | --- |
| **又名** | **Smoothing Filters (平滑滤波器)** | **Difference / Sharpening Filters (差分/锐化滤波器)** |
| **基本目的** | **Pass (通过) 低频信号**，Block (阻挡) 高频信号。 | **Pass (通过) 高频信号**，Block (阻挡) 低频信号。 |
| **空间域效果** | Smoothing (平滑), Blurring (模糊)。 | Sharpening (锐化), Edge Detection (边缘检测)。 |
| **对图像细节的影响** | **抑制或去除details (细节) 和 noise (噪声)。** | **增强details (细节) 和 edges (边缘)（同时也会放大 noise）。** |
| **Kernel (核) 特性** | **Coefficients (系数) 均为正值。** | **Coefficients (系数) 有正有负。** |
| **Kernel (核) 系数总和** | **总和通常为 1。**  **(为了在平均亮度的同时保持brightness (亮度) 不变)。** | **总和通常为 0。**  **(使其在flat (平坦) 区域的响应为0)。** |
| **主要应用** | Noise Reduction (噪声抑制)。  Image Blurring (图像模糊)。  Canny (坎尼) 边缘检测的第一步。 | Edge Detection (边缘检测)。  Image Sharpening (图像锐化)。  提取图像中的 features (特征)。 |
| **课程中的示例** | **• Box Filter (盒式滤波器) / Mean Filter (均值滤波器)。**  **• Gaussian Filter (高斯滤波器)。** | **• 一阶导数 (First-Order): Prewitt, Sobel, Roberts。**  **• 二阶导数 (Second-Order): Laplacian (拉普拉斯)。**  **• 锐化过程: Unsharp Masking (USM) (锐化掩模)。** |

**图像处理算子 (Image Processing Operators) 总结**

我把它们分为三大类：

1. **Linear Filters (线性滤波器)** (基于Convolution (卷积) / Correlation (相关))
2. **Non-Linear Filters (非线性滤波器)**：中值滤波
3. **Morphological Operators (形态学算子)** (这是一类特殊的non-linear (非线性) 算子)

**1. Linear Filters (线性滤波器)**

这类滤波器都是通过convolution (卷积) 或correlation (相关)（即邻域的weighted sum (加权求和)）来实现的。

| **滤波器/算子** | **Kernel (核) (它是什么)** | **Purpose (用来干什么)** |
| --- | --- | --- |
| **Mean / Box Filter** (均值/盒式滤波器) |  | **Smoothing (平滑) / Blurring (模糊)**。对邻域取平均值，属于low-pass filter (低通滤波器) |
| **Gaussian Filter** (高斯滤波器) |  | **Smoothing (平滑) / Blurring (模糊)**。更平滑的low-pass filter (低通滤波器)，权重由中心向外递减。Canny (坎尼) 边缘检测的第一步 。 |
| **1st-Order Derivatives** (一阶导数算子) |  | **Edge Detection (边缘检测)**。寻找gradient magnitude (梯度幅值) 的peak (峰值)。 |
| **Sobel Operator** (索贝尔算子) |  | Edge Detection (边缘检测)。结合了differentiation (微分) 和smoothing (平滑)，对噪声比 Prewitt (普鲁伊特) 更robust (鲁棒)。 |
| **Prewitt Operator** (普鲁伊特算子) |  | Edge Detection (边缘检测)。结合了differentiation (微分) 和mean filtering (均值滤波)。 |
| **Roberts Operator** (罗伯茨算子) |  | Edge Detection (边缘检测)。使用 $2 \times 2$ 窗口的对角线差分来近似梯度。 |
| **2nd-Order Derivative** (二阶导数算子) |  | Edge Detection (边缘检测) 和 Sharpening (锐化)。 |
| **Laplacian Operator** (拉普拉斯算子) |  | 1. **Edge Detection (边缘检测)**: 通过寻找其响应的 **zero-crossing (零交叉)**。***系数和为0***  2. **Image Sharpening (图像锐化)**: 将Laplacian (拉普拉斯) 结果（按一定权重）从原图中**减去**。 |
| **Laplacian of Gaussian (LoG)** (高斯-拉普拉斯) |  | Edge Detection (边缘检测)。先用Gaussian (高斯) 平滑图像，再应用Laplacian (拉普拉斯)。这解决了Laplacian (拉普拉斯) 对噪声敏感的问题。 |

**2. Non-Linear Filters (非线性滤波器)**

这类滤波器不使用weighted sum (加权求和)，而是基于ranking (排序) 或其他非线性操作。***椒盐噪声是指* 随机出现的纯白(255)或纯黑(0)像素，其余像素完好。**

| **滤波器/算子** | **Operation (操作) (它是什么)** | **Purpose (用来干什么)** |
| --- | --- | --- |
| **Median Filter** (中值滤波器) | 对kernel (核) 邻域内的所有像素值进行sort (排序)，并取\*\*median (中位数)\*\* 作为新值。 | **Noise Removal (噪声去除)**。对 ***salt and pepper noise (椒盐噪声)*** 有奇效，且比linear filters (线性滤波器) 更能**保留边缘** 21。 |
| **Min Filter** (最小值滤波器) | 取kernel (核) 邻域内的\*\*minimum (最小值)\*\* 作为新值。 | 移除bright spots (亮点)（“盐”噪声）。使暗部区域扩大 23。 |
| **Max Filter** (最大值滤波器) | 取kernel (核) 邻域内的\*\*maximum (最大值)\*\* 作为新值。 | 移除dark spots (暗点)（“胡椒”噪声）。使亮部区域扩大。 |

**3. Morphological Operators (形态学算子)**

这类算子（来自 lecture-6-morph.pdf）专门用于处理图像的\*\*shape (形状)\*\*，通常在binary image (二值图像) 上操作。

| **算子** | **Operation (操作) (它是什么)** | **Purpose (用来干什么)** |
| --- | --- | --- |
| **Dilation** (膨胀) | 使用Structuring Element (SE) (结构元素) kernel (核) 来“扩大”foreground (前景) 26。 | **Growing features (增长特征)**，**Filling holes and gaps (填充孔洞和间隙)** 27。 |
| **Erosion** (腐蚀) | 使用SE (结构元素) kernel (核) 来“收缩”foreground (前景)。 | **Shrinking features (收缩特征)**，Removing small objects (移除小物体)，Removing bridges (去除桥接)。 |
| **Opening** (开运算) | **先 Erosion (腐蚀)，后 Dilation (膨胀)**（使用同一个SE (结构元素)）30。 | Smooth contour (平滑轮廓)，**Eliminate thin protrusions (消除细突起)**，Break narrow bridges (断开窄桥) 31。 |
| **Closing** (闭运算) | **先 Dilation (膨胀)，后 Erosion (腐蚀)**（使用同一个SE (结构元素)）32。 | Smooth contour (平滑轮廓)，**Eliminate small holes (消除小孔洞)**，Fill narrow breaks and gaps (填充窄缝) 33。 |

***都是 先操作目的再找补：开-针对外部-腐蚀突起-Dilation变回原样***

**## 4. Convolution的数学性质**

该操作满足：

- 交换律：$I\*H=H\*I$

- 线性：$(s\cdot I)\*H = I\*(s\cdot H) = s \cdot (I\*H)$，$(I\_1+I\_2)\*H = I\_1 \* H + I\_2 \*H$

- 结合律：$A\*(B\*C) = (A\*B)\*C$

- 不满足数和卷积的结合：$(b+I)\*H\ne b+(I\*H)$​

- 可分离性：$I\*H = I\*(H\_1 + \cdots H\_n) = (((I\*H\_1)\*H\_2)\*\cdots)\*H\_n$

**## 5. Impulse (Dirac) Function： 脉冲函数（狄拉克函数）**

* 把 δ 看成“无限高、无限窄、面积为 1”的尖钉。
* 现实中可用窄脉冲+高幅值逼近，脉冲越窄、越高，保持面积=1。

- 转换为图片：原点像素值为白色，其余部分黑色

- 卷积不变：该函数作为filter时，无法改变图像

- 卷积作用于该函数时，该函数直接变为filter

**## 6. Non-Linear Filter**

**### 6.1 Noise**

在成像时，由于各种原因，图像上总会出现不想要的噪点，因此需要进行各种去躁技术。

> 噪音的种类

> 1. 椒盐噪声Salt and Pepper: 随机出现的黑色/白色像素点

> 2. Gaussian噪声：服从正态分布的噪声，对白噪声的近似

>    $I+Noise,\quad Noies \sim N(\mu, \sigma)$

> 3. Speckle噪声：$I\times(1+Noise)$，Noise为随机噪声

> 4. Periodic周期性噪声

> 前三种可以通过空间滤波消除，最后一种需要使用频域滤波消除

**### 6.2 Non-Linear Filters**

与线性滤波不同，非线性滤波使用了非线性函数来组合区域内的像素值

**#### 6.2.1 Min / Max Filter**

取filter区域内最小/最大像素值作为中心像素的值。

如果在filter宽度内，信号剧烈变化，则会将该变化滞后，如图a；如果有剧烈的抖动，则会消除抖动，如图b；如果信号线性增加，则会让整个信号增加趋势向右移动，如图c。

使用min-max filter可以消除椒盐噪声，即使用max filter消除白色噪点，使用min filter消除黑色噪点。

**#### 6.2.2 Median Filter**

中值滤波器，取filter区域所有像素值的中位数作为新的像素值。效果如下

中值滤波器可以：1. 消除孤立点（消除椒盐噪声）；2.移除细线；3. 保留边缘；4. 平滑转角

**#### 6.2.3 Weighted Median Filter**

带权中值滤波器，使用一个权重矩阵$W(i,j)$，对应位置表示点(i,j)在最终排列中的出现次数，再从排列中取中位数作为目标像素值。

**# IV. Edge and Contours**

**## 1. Edge**

边缘，通常是指图像中亮度变化剧烈的部分（即不连续）。可能是由于：1.物体实际边界；2. 光照的不均匀（例如反射、表面材质等）。边缘检测是图像处理中的一个重要技术，用于提取图像中物体的边缘与轮廓信息。

**### 1.1 边缘的特性**

- 亮度的突然变化：一个理想化的边缘应该是某一方向上像素值相对于像素位置的阶跃函数

- 真实情况下，边缘通常是一个近似阶跃函数。可以通过较大的一阶导数值来确定边缘。

- 边缘处的像素值变化：一阶导数达到最值，二阶导数是跨轴零点

- 图片的像素变化通常是离散函数，因此考虑其一阶差分

  - 离散函数$f(u)$​在点u处的一阶差分被定义为（central difference）

# Corner Detection

在许多计算机视觉任务中，转角Corner被认为是一个图像中较为鲁棒的特征，据此可以完成诸多匹配任务。

我们刚刚学过的**edge (边缘) 是一种 feature (特征)，但它只在一个方向上是明确的（垂直于边缘的方向），而在沿着边缘的方向上是模糊的** 。

**Corners (角点) 是一种更好的 keypoint (关键点)，因为它们在所有方向上都是明确的**，这使它们成为图像matching (匹配) 和tracking (跟踪) 的理想选择 。

**1. 为什么角点是好的特征？(Why are Corners Good Features?)**

想象一下，你要在两张图片中match (匹配) 同一个点。如 lecture-7-corner-line.pdf 第29-30页所示：

* 如果你选一个\*\*flat (平坦)\*\* 区域（比如一面白墙）的patch (小块)，它在另一张图中会有无数个相似的匹配项，你无法确定是哪一个。
* 如果你选一个\*\*edge (边缘)\*\* 上的patch (小块)，它在另一张图中沿着边缘滑动时，看起来都一样。你只能确定它在这条线上，但不知道具体在哪里。
* 但如果你选一个\*\*corner (角点)\*\* 上的patch (小块)，它在两个方向上都有剧烈的intensity change (强度变化)。这个patch (小块) 是非常unique (独特) 的，在另一张图中只有一个最好的匹配。

**2. Harris Corner Detector (Harris角点检测器)**

Harris (哈里斯) 角点检测器是一种经典的算法，它用数学方法来**找到这些“在所有方向上移动窗口都会产生巨大强度变化”的点**。

**A. 核心思想 (The Math)**

1. 我们想知道：如果我们把一个小的patch (小块) 或window (窗口) 在 $(u,v)$ 方向上移动一点点，intensity (强度) 会变化多少？
2. 这个变化可以用\*\*Sum of Squared Differences (SSD) (差值平方和)\*\* 来衡量，如 lecture-7-corner-line.pdf 第40页 7 所示：
   * $I(x,y)$ 是原始强度。
   * $I(x+u, y+v)$ 是移动 $(u,v)$ 后的强度。
   * $w(x,y)$ 是一个window function (窗口函数)（比如 Gaussian (高斯)），确保我们只关心这个小patch (小块) 内部。
3. 当 $(u,v)$ 很小时，通过Taylor Series Expansion (泰勒级数展开)，这个复杂的 $E(u,v)$ 公式可以被近似为一个简单的二次型：
4. 这里的 $M$ 就是关键，它是一个 $2 \times 2$ 矩阵，通常被称为“structure tensor (结构张量)”。它是通过图像derivatives (导数) $I\_x$ 和 $I\_y$ 计算得来的：

**B. 分析 $M$ 矩阵 (Analyzing the Matrix M)**

这个 $M$ 矩阵包含了patch (小块) 内部所有gradient (梯度) 信息的统计。lecture-7-corner-line.pdf 第52页 的图解非常重要：

这个矩阵 $M$ 的 **eigenvalues (特征值)**，我们称之为 $\lambda\_1$ 和 $\lambda\_2$，完美地描述了我们正在看的patch (小块) 类型：

1. **Flat (平坦) 区域:**
   * 梯度在所有方向上都接近0。
   * $\lambda\_1 \approx 0$ 且 $\lambda\_2 \approx 0$ (两个特征值都**很小**)
2. **Edge (边缘):**
   * 梯度只在**一个**方向（垂直于边缘）上很大。
   * $\lambda\_1 \gg 0$ 且 $\lambda\_2 \approx 0$ (一个特征值**很大**，一个特征值**很小**)。
3. **Corner (角点):**
   * 梯度在**多个**方向上都很大。
   * $\lambda\_1 > 0$ 且 $\lambda\_2 > 0$ (两个特征值都**很大**) 13。

**C. Harris Response Function (Harris 响应函数) $R$**

计算eigenvalues (特征值) $\lambda\_1, \lambda\_2$ 的开销很大。Harris 提出了一个更快的计算方法，即 **Harris Response Function (Harris 响应函数) $R$**，它不直接计算特征值，而是使用矩阵的\*\*determinant (行列式)\*\* 和 **trace (迹)** 来近似判断：

其中:

* $\alpha$ 是一个经验常数 (通常取 0.04 到 0.06) 15。

现在，我们可以根据 $R$ 的值来分类像素了（如 lecture-7-corner-line.pdf 第56页所示）：

1. **Corner (角点):** $R$ 是一个**大的正数** (因为 $\lambda\_1 \lambda\_2$ 很大, 且 $\lambda\_1 + \lambda\_2$ 相对较小)。
2. **Edge (边缘):** $R$ 是一个**负数** (因为 $\alpha (\lambda\_1 + \lambda\_2)^2$ 项远大于 $\lambda\_1 \lambda\_2$ (此时 $\lambda\_2 \approx 0$) )。
3. **Flat (平坦):** $R$ 的**绝对值很小** (接近0)。

**3. Harris Detector (Harris 检测器) 的完整算法**

1. 计算图像在 $x$ 和 $y$ 方向的导数 $I\_x, I\_y$ （例如使用 Sobel 算子）。
2. 计算 $I\_x^2, I\_y^2, I\_x I\_y$。
3. 使用 Gaussian filter (高斯滤波器) 对这三个值进行smoothing (平滑)，得到
4. 对每个像素，计算 Harris response (Harris 响应) 。
5. **Thresholding (阈值处理):** 找到所有 $R > t\_H$ 的点（$t\_H$ 是一个设定的阈值）。
6. **Non-Maxima Suppression (非极大值抑制):** 在一个小的邻域内（比如 $3 \times 3$），只保留 $R$ 值最大的那个点，以确保一个角点只被检测到一次 20。

**一个重要特性：** Harris (哈里斯) 检测器**具有\*\*rotation invariance (旋转不变性)（旋转图像，角点仍然会被检测到），但不具有scale invariance (尺度不变性)\*\*** 21。如果你scale (缩放) 图像，corner (角点) 可能会变成 edge (边缘) 或 flat (平坦) 区域，反之亦然。

按照考纲，下一部分是 **Curve Detection (曲线检测)**（在 lecture-7-corner-line.pdf 中也有涉及）和 **Morphological Filters (形态学滤波器)**（在 lecture-6-morph.pdf 中）。

**## 2. Corner**

转角，通常被视为一种鲁棒的图片信息，是轮廓的交汇处。在图像处理中，转角一般表现为沿不同方向，像素强度变化的差异较大。

**### 2.1 Corner Point**

从patch matching的角度来说，转角点Corner Point通常有如下特征：当patch窗口位于转角点时，无论向任何方向移动都会造成像素强度的剧烈变化。

判断一个patch是否落在corner处的方法，其思想是：向不同方向移动该patch的窗口，如果像素强度发生了剧烈变化，则该patch位于转角处。

**### 2.2 Harris Corner Detector**

**#### 2.2.1 基本思想**

一个patch在不同区域的表现：

1. 在普通平整区域时，像素值在任何方向上都不会有剧烈的变化；
2. 在边缘区域时，像素值沿着边缘区域不会产生剧烈变化；
3. 在转角区域时，在许多方向上都会产生剧烈的变化。

**Harris转角检测方法，就是定量地使用数学方法以确定像素变化属于上述哪种情况，从而实现转角检测。**

**#### 2.2.2 数学方法**

Harris方法的核心思想是：计算一个patch从像素点(x,y)向各个方向上移动(u,v)个像素后的像素值变化，

- 对于一个平整区域，上式中的平方距离不会很大

- 对于转角区域，E(u,v)的值会较大

进一步地，对该平方距离进行Taylor展开并保留一阶项：

$I(x+u,y+v)-I(x,y)=I(x,y)+uI\_x+vI\_y-I(x,y)=uI\_x+vI\_y$

其中$I\_x,I\_y$表示对x和y求一阶偏导，其平方即：

因此，对于一个微小的(u, v)变化，E(u,v)就可以近似为：

> 其中**$\lambda\_1, \lambda\_2$为特征值**，即：

获得了特征值后，观察特征值，即可判定patch所在区域的情况：

***- 平整区域：两个特征值都较小***

***- 边缘区域：其中一个特征值远大于另一个特征值***

***- 转角区域：两个特征值都较大且几乎相同***

这两个特征值一个对应其中一条边(+)，另一个对应于其中另一条边(-)，对于一个转角，第二条边的特征值（-）应该同样具有显著的大小。在实践中，可以使用如下Harris Corner Response函数来进行判断

> - 函数值仅取决于特征值$\lambda\_1,\lambda\_2$

> - 函数值在转角处较大

> - 函数值在边缘处出现较大的负值

> - 在平整区域，函数值绝对值较小

## Curve Detection and Morphological Filters

我们刚刚学完Edge detection (边缘检测)，它给了我们一堆零散的edge pixels (边缘像素) 1。现在的任务是，如何从这些乱七八糟的点中，找出有意义的“全局”形状，比如lines (直线) 和circles (圆)。

这就是 **Hough Transform (霍夫变换)** 的用武之地。

**1. 核心思想：从“图像空间”到“参数空间”**

Hough Transform (霍夫变换) 2的核心思想是一种voting (投票) 机制。它巧妙地将问题从 **Image Space (图像空间)**（我们平时看到的 $(x, y)$ 坐标系）转换到了一个 **Parameter Space (参数空间)**（一个由形状参数组成的坐标系）3。

我们先以最简单的**直线**为例。

**2. 直线检测 (Line Detection)**

**A. 参数空间：$y = kx + d$**

在Image Space (图像空间) 中，一条直线由两个参数定义：slope (斜率) $k$ 和intercept (截距) $d$ 4。所以，我们的Parameter Space (参数空间) 就是一个以 $k$ 为横轴、$d$ 为纵轴的坐标系。

**B. 关键的“对偶性” (Duality)**

Hough Transform (霍夫变换) 的“魔法”在于Image Space (图像空间) 和Parameter Space (参数空间) 之间的“对偶”关系 6：

1. **Image Space (图像空间) 中的一个点 $\rightarrow$ Parameter Space (参数空间) 中的一条线**
   * Image Space (图像空间) 中有一个点 $(x\_1, y\_1)$。有**无数条**不同 $(k, d)$ 的直线可以穿过这个点 7。
   * 所有这些能穿过 $(x\_1, y\_1)$ 的直线都满足 $y\_1 = kx\_1 + d$。
   * 如果我们把 $k$ 和 $d$ 看作变量，这个方程就变成了 $d = (-x\_1)k + y\_1$。这在 (k, d) 参数空间中，是一条斜率为 $-x\_1$、截距为 $y\_1$ 的**直线**。
2. **Image Space (图像空间) 中的一条线 $\rightarrow$ Parameter Space (参数空间) 中的一个点**
   * Image Space (图像空间) 中有一条**特定**的直线，比如 $y = 2x + 1$。
   * 它在Parameter Space (参数空间) 中对应**一个点**，即 $(k=2, d=1)$。

**C. “投票”过程 (The Voting Process)**

Hough Transform (霍夫变换) 就是利用这个对偶性来进行“投票” 10：

1. 我们创建一个2D网格，叫作 **Accumulator Array (累加器数组)**，它就是离散化的Parameter Space (参数空间) $(k, d)$ 11。
2. 我们遍历Edge Map (边缘图) 上的**每一个**edge pixel (边缘像素) $(x\_i, y\_i)$。
3. 每找到一个edge pixel (边缘像素)，我们就在Parameter Space (参数空间) 中画出它对应的那条直线 $d = (-x\_i)k + y\_i$。这条直线穿过的所有accumulator (累加器) 单元格，票数都+1。
4. **关键：** 如果Image Space (图像空间) 中有**3个**edge pixel (边缘像素) **在同一条直线**上，那么它们在Parameter Space (参数空间) 中对应的**3条线**必然会**相交于同一个点**。
5. 所有edge pixel (边缘像素) 都投完票后，我们就在Accumulator Array (累加器数组) 中寻找**得票最高**的那个点 $(k\_{peak}, d\_{peak})$ 13。
6. 这个“票王” $(k\_{peak}, d\_{peak})$ 就代表了Image Space (图像空间) 中那条**拥有最多edge pixel (边缘像素) 的直线** 14。

**D. (k, d) 空间的问题与解决**

* **问题：** y = kx + d 这种参数表示法，无法表示\*\*vertical line (垂直直线)\*\*，因为此时slope (斜率) $k$ 为infinity (无穷大)。
* 解决方案： 使用 polar coordinates (极坐标) 来重新参数化直线。
  + **$** **$ (rho):** 原点到直线的垂直距离。
  + **$ $ (theta):** 该垂直线的角度。
* **新的参数空间：** 现在是 $(\rho, \theta)$。
* **新的对偶性：** Image Space (图像空间) 中的一个点 $(x, y)$，在 $(\rho, \theta)$ 空间中对应一条\*\*sinusoidal curve (正弦曲线)\*\*： $\rho = x \cos \theta + y \sin \theta$。
* **投票：** 投票过程完全一样。Image Space (图像空间) 中共线的点，在 $(\rho, \theta)$ 空间中对应的正弦曲线会相交于一点 18。

Hough Transform (霍夫变换) 是一个非常巧妙的**从local features (局部特征)（边缘点）中检测global shapes (全局形状)（直线）**的方法。

lecture-7-corner-line.pdf 的第22页 19还介绍了如何将其扩展到\*\*Circle Hough Transform (圆形霍夫变换)\*\*，原理是完全一样的，只是Parameter Space (参数空间) 变成了 $(x\_0, y\_0, r)$（圆心坐标和半径）20。

**### 1.5 噪音的处理**

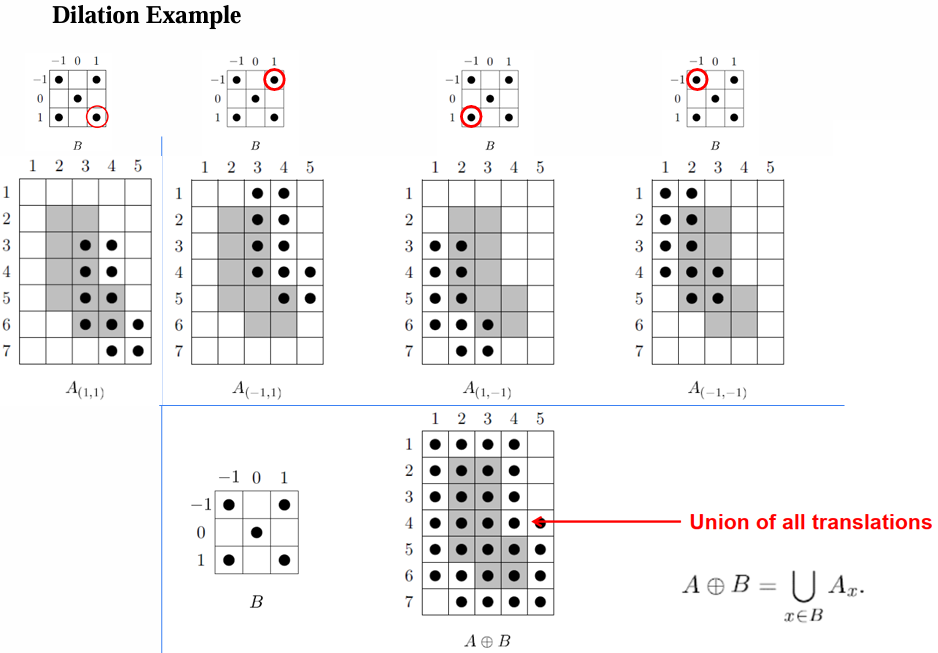
在这幅表示Hough Transform的图中，图a的像素空间四条直线在图b的参数空间中反应为四个最密集的交汇点。然而图片中同样存在许多噪点，即最大的$r-\theta$值有时出现在一片区域内而非一个点上，并且由于$\theta$的离散化会导致难以精确定位最大值。对此有两种解决办法：**设置阈值与非极大值抑制**。

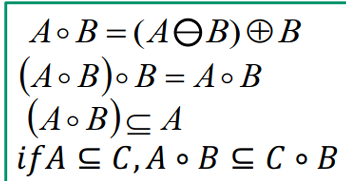
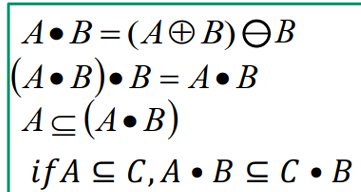
- 设置阈值：只保留50%极大值

- 非极大值抑制：将每个候选点都与周围的点比较，只保留最大的

## Morphological Filters





**1. 什么是 Morphological Processing (形态学处理)？**

Morphological (形态学) 这个词源于生物学，意思是研究物体的“**shape (形状)**”和“**form (形态)**”。

* Linear Filters (线性滤波器)（如Gaussian (高斯)）是基于convolution (卷积) 的weighted sum (加权求和) 来操作的。
* Morphological Processing (形态学处理) 则完全不同。它是一种non-linear (非线性) 的操作，它通过一个“探针”去探测图像，并根据图像的“形状”来修改它 1。

这些操作在 binary image (二值图像)（只有0-黑 和 1-白）上最容易理解，它们是进行图像segmentation (分割)、feature extraction (特征提取) 和noise removal (噪声去除) 的强大工具。

**2. 核心工具：Structuring Element (SE) (结构元素)**

Morphological operations (形态学操作) 的kernel (核) 不叫kernel (核)，而叫 **Structuring Element (SE) (结构元素)**。

* **它是什么？** SE (结构元素) 是一个小型的binary shape mask (二值形状掩码)，你可以把它想象成一个“探针”或“邮票”。
* **它长什么样？** 它可以是任何形状，比如 square (正方形)、cross (十字形) 5、disk (圆形) 6或你定义的任何形状。
* **它有什么？** 它有一个origin (原点)，也叫hot spot (热点)，用来在kernel (核) 在图像上滑动时进行对齐 8。

***所有形态学操作，都是基于这个SE (结构元素) 与图像的neighborhood (邻域) 进行交互（比如“它是否能放得下？”或“它是否碰到了？”）来定义的***。

**3. 两个最基本的操作：Dilation (膨胀) 和 Erosion (腐蚀)**

这是所有形态学操作的基础 9。

**A. Dilation (膨胀) $A \oplus B$**

* **直观理解：** Growing (生长) 或 Expanding (扩张) 10。
* **作用：** Fill holes and gaps (填充孔洞和间隙) 11。
* 如何工作 (Set Theory):

lecture-6-morph.pdf 第26页的定义是 $A \oplus B = \bigcup\_{x \in B} A\_x$。

一个更直观的理解是 13：**并集**：在Image (图像) $A$ 中的每一个foreground (前景) 像素上，都“盖”一个SE (结构元素) $B$ 的“邮票”。所有这些“邮票”区域的Union (并集) 就是Dilation (膨胀) 的结果。

* **结果：** foreground (前景) 物体（白\*\*\*域）会变大、变胖。小的黑色孔洞会被填满。

**B. Erosion (腐蚀) $A \ominus B$**

* **直观理解：** Shrinking (收缩) 或 Eroding (侵蚀)。
* **作用：** Removing small objects (移除小的物体/噪声)，Removing bridges (断开细小的连接)，以及separating blobs (分离粘连的物体)。
* 如何工作 (Set Theory):

lecture-6-morph.pdf 第37页 16 的定义是 $A \ominus B = \{w : B\_w \subseteq A\}$。

这一定义是关键：**交集**：只有当SE (结构元素) $B$（以 $w$ 为中心）fits entirely inside (完全包含)\*\* 在Image (图像) $A$ 的foreground (前景) 区域时，output (输出) 图像在 $w$ 处的像素才被设为1（前景）17。

* **结果：** foreground (前景) 物体（白色区域）会变小、变瘦。细小的连接会被“腐蚀”断，小的噪声点会完全消失。

**4. 组合操作：Opening (开运算) 和 Closing (闭运算)**

Dilation (膨胀) 和Erosion (腐蚀) 是“可逆”的吗？不是。

Erosion (腐蚀) 之后再Dilation (膨胀)，并不能恢复原状。但这组操作却有一个非常有用的名字：Opening (开运算)。

**A. Opening (开运算) $A \circ B$**

* **定义：** 先 **Erosion (腐蚀)**，后 **Dilation (膨胀)**。
* **直观理解：** 想象 SE (结构元素) 像一个球，在foreground (前景) 物体的\*\*inside (内部)\*\* 边界“滚动” 20。它会“扫平”所有它进不去的thin protrusions (细小突起)。
* **作用：** Eliminate thin protrusions (消除细小突起) 和 Break narrow bridges (断开窄桥) 21。它会**打开**物体间的小缝隙。

**B. Closing (闭运算) $A \bullet B$**

* **定义：** 先 **Dilation (膨胀)**，后 **Erosion (腐蚀)**。
* **直观理解：** 想象 SE (结构元素) 像一个球，在foreground (前景) 物体的\*\*outside (外部)\*\* 边界“滚动” 23。它会“填满”所有它掉不进去的gaps (缝隙) 和holes (孔洞)。
* **作用：** Fill narrow breaks and gaps (填充窄缝) 和 Eliminate small holes (消除小孔洞) 24。它会**闭合**物体内部的小洞。

**Morphological Processing (形态学处理)** 的核心就是这四种操作

**## 2.**

**### 2.1 Mathematical Morphology**

该技术最初是在二值化（黑白）图像上进行操作的

- **\*\*Translation\*\*** 记A为黑白图像上的某些像素点集合，其中w = (x, y)是一个特定坐标点，$A\_s$被定义为由A向（x, y）方向的Translation：

- **\*\*Reflection\*\***  将集合A的所有点关于原点中心对称：

***- \*\*Dilation\*\*  扩展相连的1集合，可以用于成长features以及填补缝隙***

***- \*\*Erosion\*\*  缩减相连的1集合，可以用于缩减features以及删除连接与凸起***

- **\*\*Shrink与Grow\*\***： shrink是指不断去掉边界像素点的过程，grow是指不断向边界添加像素点的过程

**### 2.2 Basic Morphological Operations**

**#### 2.2.1 Dilation**

设A和B为两个像素点集合，则dilation of A by B被定义为：

即将A沿B中各个点的方向做Translation，从而形成的总覆盖面积。

注意到：1. Dilation操作可以扩展A结构的面积；2. A和B不一定需要有重合部分

对一个黑白图片使用Dilation操作的效果：

在这一过程中，通常将A设定为待处理的结构，而B通常为一个较小的集合，被称为structuring elements

> **\*\*structuring elements\*\***

> 结构元素，通常被用作形态操作中的mask，其形状和大小随意，具有原点并且每个点可以用坐标表示，通常具有以下特征：

> - 类似于filter

> - 只有0，1两个元素值

**#### 2.2.2 Erosion**

设A和B为两个像素点集合，则dilation of A by B被定义为：

即对A中每个点沿着B中每个点的方向做translation，取所有集合的交集。直观理解，即找到A中所有出现B的位置

> **#### Dilation 与 Erosion 的性质**

> - 二者互逆

> - Dilation满足结合率、交换律

> - Erosion不满足交换律、结合率（类似减法，需变号）

**### 2.4 Morphological Filters**

参考structuring element B，可以设计形态学滤波器，该滤波器由如下两部分定义：

## Color Image Processing

**## 1. What is Color?**

颜色有不同种类的描述方式，例如光的波长；视神经冲动信号等。人眼可以感知到波长380nm至720nm的光对应的颜色。

**## 2. Color Space**

**### 2.1 RGB**

RGB是最常见的色彩空间，由红Red绿Green蓝Blue三通道组成，其中(0, 0, 0)表示黑色，(255, 255, 255)表示白色。几乎所有在灰度图上的操作都可以对RGB图像的三个通道分别进行。

**### 2.2 CMY(K)**

CMY(K)常用于印刷领域，由青色Cyan，洋红色Magenta与黄色Yellow构成，有时也会加入黑色（K）以达到更好的效果。与RGB叠加通道不同，(C, M, Y)表示将三个通道相减。

**### 2.3 HLS**

基于色相(Hue)，明度(Lightness)，饱和度(Saturation)的色彩空间，是对RGB正方体的扭曲。

**### 2.4 HSV**

基于色相(Hue)，饱和度(Saturation)，亮度(Value, Brightness)定义的色彩空间。

**## 3. 色彩转换**

**### 3.1 色彩空间互相转换**

色彩空间的互相转换可以视为一个矩阵变换：

**### 3.2 彩色图像转为灰度图**

- 可以使用一个简单转化，即对三通道取平均值：

  但这样做会导致R区和G区的颜色过暗。

- 可以通过赋予三个通道不同的权重来进行转换：

  一般来说，权重$w\_G > w\_R > w\_B$。

- Hueless Color Image：当RGB图像的三个通道像素分量相等时，称为灰度图像（或Hueless）。

**## 4. Organization of Color Images**

一般来说，根据对色彩的运用，可以颜色分为True Color或Indexed Color。

**### 4.1 True Color**

使用了色彩空间中的所有颜色，常用于需要分辨色彩之间细微差别的应用领域，例如数码摄影等。True color图片的组织有两种方式：component ordering与packed ordering

- Component Ordering：将不同色彩通道分别存储成二维数组，使用时从三个通道中恢复色彩数据。

- Packed ordering：直接在一个pack中存储RGB三个数据。

**### 4.2 Index Color Images**

这种类型的图片仅含有有限种类的颜色（N=2 to 256），常用于色彩重复度较高的图像中。在该类型图像中，像素点存储的数据是对应颜色的索引位置，同时存储一个该图片中所有颜色的列表。

**## 5. 处理彩色图片**

处理彩色图片一般有两种策略：单独处理每个RGB矩阵，或先计算其加权平均后处理像素强度矩阵。

**### 5.1 RGB to HSV**

**1. 什么是 Color Image (彩色图像)？**

首先，我们回顾一下 lecture-1-intro.pdf 第20页和第21页 1 的内容。

* Grayscale (灰度) 图像：是一个2D数组 $f(x,y)$。每个pixel (像素) 只有一个value (数值)（代表亮度）。
* Color (彩色) 图像：是一个3D数组。每个pixel (像素) $(x,y)$ 包含**三个**values (数值)，分别代表三个channels (通道)。
* 最常见的模型是 **RGB** 2：
  + **Red R[x,y]**
  + **Green G[x,y]**
  + **Blue B[x,y]**
* **Point Operations (点操作)** 可以直接应用于彩色图像，通常是**独立地** (independently) 应用于 R, G, B 三个通道。例如，Image Negative (图像反转) 就是：
  + $R' = 255 - R$
  + $G' = 255 - G$
  + $B' = 255 - B$

**2. Color Histograms (彩色直方图)**

lecture-3-hist.pdf 的第12页和第13页探讨了我们如何为彩色图像计算histogram (直方图)。主要有两种方法：

1. **Intensity Histogram (亮度直方图):**
   * 首先，将color image (彩色图像) 转换为grayscale (灰度) 图像。
   * 然后，只计算这个grayscale (灰度) 图像的histogram (直方图)。
   * **缺点：** 这种方法**完全丢失了颜色信息** 5。
2. **Individual Channel Histograms (独立通道直方图):**
   * 计算**三个**独立的histograms (直方图)：一个Red (红色) 通道的，一个Green (绿色) 通道的，一个Blue (蓝色) 通道的。
   * **缺点：** 这种方法虽然保留了颜色统计，但它**丢失了颜色之间的相关性**。它告诉你“有多少红色像素”和“有多少绿色像素”，但不能告诉你“有多少黄色像素”（即红色和绿色都很高的像素）7。

**3. RGB 模型的局限性**

在 RGB 空间中进行图像处理（比如增强对比度）是非常困难的。

* **问题：** Luminance (亮度) 和 Chrominance (色度/颜色信息) **are mixed together (混合在一起)**。
* **例子：** 假设你想让一张图片“变亮”。如果你只是简单地把 R, G, B 的值都增加，你可能会无意中改变它的color (颜色)（例如，一个深红色可能会变成亮粉色）。

**4. 解决方案：分离 Luminance (亮度) 和 Chrominance (色度)**

为了解决这个问题，我们使用**感知型** (perceptually-based) 的颜色空间，将\*\*Luminance (亮度)\*\*（我们对它最敏感）与 **Chrominance (色度)**（我们对它不太敏感）分离开。

lecture-9-part2-compression\_v2.pdf 在讲解 JPEG (JPEG压缩) 时，详细介绍了这个概念。JPEG (JPEG压缩) 流程的第一步就是 **Color-Space Convertor (色彩空间转换)**：

* **输入：** RGB
* **输出：** YUV (或类似的 YCbCr)
  + **Y:** Luminance (亮度) 通道（即黑白图像）。
  + **U, V:** Chrominance (色度) 通道（即颜色信息）。

**5. 为什么分离是有效的？Contrast Sensitivity (对比度敏感度)**

lecture-9-part2-compression\_v2.pdf 的第25-30页 解释了为什么要这样做：

**人类的视觉系统对Luminance (亮度) 的high-frequency (高频) 细节非常敏感，但对Chrominance (色度) 的high-frequency (高频) 细节**less sensitive (**不敏感**)。

* 看第29页 的对比图：
  + 在下半部分的Luminance (亮度) 图中，即使条纹很密（高频），你也能看清。
  + 在上半部分的Chrominance (色度) 图中（红-绿条纹），当条纹变密时，它们很快就**糊在一起**，你看不清细节了。

**6. 应用：Chroma Subsampling (色度子采样)**

JPEG (JPEG压缩) 和MPEG (MPEG压缩) 等压缩技术利用了人类视觉的这个“弱点”来实现高效压缩，这个技术叫做 **Chroma Subsampling (色度子采样)**。

* **思路：** 既然我们对Luminance (亮度) 细节这么敏感，那我们就**完整保留** $Y$ 通道。
* 既然我们对Color (颜色) 细节不敏感，那我们就可以**大量丢弃** $U$ 和 $V$ 通道的信息。
* **操作：** 如 lecture-9-part2-compression\_v2.pdf 的第20-23页 所示，我们可以在一个 $4 \times 4$ 的像素块中，只存储 $Y$ 通道的全部16个像素，但**只存储1个** $U$ 像素和**1个** $V$ 像素。
* **结果：** 仅这一步，数据量就减少了 **62.5%**，而人眼几乎**无法察觉**到这种颜色信息的损失。

Color image processing (彩色图像处理) 的核心就是理解RGB的局限性，以及为什么使用 YUV (亮度-色度) 这种分离的空间进行处理（尤其是压缩）会更有效。

## Spectral Techniques

**1. 核心思想：空间域 vs. 频谱域**

到目前为止，我们所有的操作（点操作、滤波器、形态学）都是在 **Spatial Domain (空间域)** 中进行的。也就是说，我们直接操作图像的 $(x, y)$ 像素矩阵。

Spectral Techniques (频谱技术) 采用了一种完全不同的思路。它将图像从Spatial Domain (空间域) 转换到一个全新的维度，称为 **Frequency Domain (频率域)**。

* **Frequency (频率)** 在图像中代表什么？
  + **Low Frequency (低频):** 图像中**缓慢**变化的区域，比如大片的蓝天、平滑的墙壁（blurring (模糊) 就是保留低频）。
  + **High Frequency (高频):** 图像中**剧烈**变化的区域，比如edges (边缘)、noise (噪声)、精细的texture (纹理)（sharpening (锐化) 就是增强高频）。

**2. Fourier Transform (傅里叶变换)**

***Fourier Transform (傅里叶变换) 是连接这两个“域”的桥梁***。**用多种正弦/余弦波来表示图像。**

* **核心思想：** lecture-10-spectral.pdf 第4页 1有一个完美的图示：**任何complex function (复杂函数)（比如图像的一行像素），都可以被分解 (decomposed) 为一系列不同频率的sines (正弦) 和cosines (余弦) 函数的summation (总和)**。
* **Fourier Transform (傅里叶变换) $G(\omega)$:** 是一种数学工具，它能告诉你，要组合成你的原始信号 $g(x)$，每种频率的正弦/余弦波各需要“多少量”（即amplitude (幅值) 和phase (相位)）。
* **Inverse Fourier Transform (反傅里叶变换) $g(x)$:** 是一种逆向工具，它能把你告诉它的所有频率和“量”，重新组合回原始信号 4。
* 正向变换：空间域 → 频率域
* 逆变换：频率域 → 空间域

**3. Discrete Fourier Transform (DFT) (离散傅里叶变换)**

由于我们的digital image (数字图像) 是discrete (离散) 的 5，我们使用Discrete Fourier Transform (DFT) (离散傅里叶变换) 6。

**A. 2D DFT**

对于 $M \times N$ 的图像 $f(x,y)$，2D DFT (二维离散傅里叶变换) $F(u,v)$ 将其转换为一个 $M \times N$ 的complex number (复数) 矩阵，代表其frequency spectrum (频谱)。

* $F(u,v)$ 中的每个点 $(u,v)$ 代表一个特定的**二维频率**（即水平和垂直方向的正弦波）。
* 这个点的值（一个复数）代表了该频率波的amplitude (幅值) 和phase (相位)。

**B. 2D DFT 的重要属性**

1. **Separability (可分离性):** 这是计算上的一个巨大优化。计算一个2D DFT，不需要一次性处理2D矩阵。你可以：
   1. 对图像的**每一行** (row) 分别进行 1D DFT。
   2. 在得到的结果上，再对每一列 (column) 分别进行 1D DFT。

这使计算速度快得多。

1. Convolution Theorem (卷积定理):

这是使用DFT最重要的原因之一！我们在 lecture-4-filter.pdf 中学到，Spatial Domain (空间域) 中的Convolution (卷积) ( $f \* h$ ) 非常慢（尤其是kernel (核) $h$ 很大时）。

Convolution Theorem (卷积定理) 告诉我们：

**空间域的 Convolution (卷积) 频率域的 Multiplication (逐元素乘法)**

所以，计算 $f \* h$ 的**更快**的方法是：

* 1. 计算 $f$ 的 DFT (傅里叶变换)，得到 $F$。
  2. 计算 $h$ 的 DFT (傅里叶变换)，得到 $H$。
  3. 在Frequency Domain (频率域) 中进行**逐元素相乘**：$G = F \cdot H$ 13。
  4. 对 $G$ 进行Inverse DFT (反傅里叶变换)，得到 $g$，这个 $g$ 就等于 $f \* h$ 14。

**当kernel (核) $h$ 很大时（例如 $32 \times 32$），这个方法的计算速度有large speedups (巨大提升)** 15。

**C. 理解 DFT 的输出 (频谱图)**

DFT (傅里叶变换) 的输出 $F(u,v)$ 不太直观。

* **DC Component (直流分量):** $F(0,0)$（矩阵的左上角）是DC component (直流分量) 16。它的值等于**整张原始图像所有像素值的总和**，代表了图像的average brightness (平均亮度)，默认做归一化。
* **Centering (中心化):** DFT (傅里叶变换) 默认把 $F(0,0)$（低频）放在左上角，而高频分布在四周。为了方便观察，我们通常会swap four quadrants (交换四个象限)，把DC component (直流分量) 移到图像**中心**。中心化后：
  + **中心 = Low Frequencies (低频)**
  + **远离中心 = High Frequencies (高频)**
* **Log Transform (对数变换):** DC component (直流分量) 的magnitude (幅度) 通常比其他所有频率都大得多。为了能看清高频细节，我们通常显示的是 $log(1 + |F(u,v)|)$ 19。

**4. Filtering in Frequency Domain (频域滤波)**

有了Convolution Theorem (卷积定理)，我们就可以在Frequency Domain (频率域) 中进行滤波了 20：

1. **Ideal Low-Pass Filter (理想低通滤波器):**
   * **做什么：** 只保留DC component (直流分量)（中心）附近的一个圆形区域内的frequencies (频率)，把圆外的所有frequencies (频率) **设为0** 21。
   * **结果：** Blurring (模糊)。圆越小，模糊越严重。
   * **问题：** 这种“理想”的sharp cutoff (锐利截止) 会在Spatial Domain (空间域) 中产生 **ringing (振铃)** 伪影（波纹）23。
2. **Ideal High-Pass Filter (理想高通滤波器):**
   * **做什么：** **设为0** DC component (直流分量)（中心）附近的一个圆形区域，保留圆外的所有frequencies (频率) 24。
   * **结果：** Sharpening (锐化) / Edge Detection (边缘检测)。圆越大，保留的高频越少，只剩下最剧烈的边缘 25252525。
3. **Butterworth Filter (巴特沃斯滤波器):**
   * **做什么：** 为了解决ringing (振铃) 问题，Butterworth Filter (巴特沃斯滤波器) 提供了一种\*\*gentle cutoff slope (平滑的截止斜坡)\*\*，而不是Ideal Filter (理想滤波器) 的“悬崖式”截止 26。
   * **结果：** 同样实现low-pass (低通) 或high-pass (高通) 滤波，但ringing (振铃) 伪影会**大大减少** 27。
4. Gaussian Filter (高斯滤波器):

Gaussian (高斯) 函数在空间域是一个low-pass filter (低通滤波器)，它在频率域**也是一个Gaussian (高斯) 函数** 28。因此，在频率域进行高斯模糊，就是将图像的DFT (傅里叶变换) 乘以一个Gaussian (高斯) 形状的mask (掩码)。

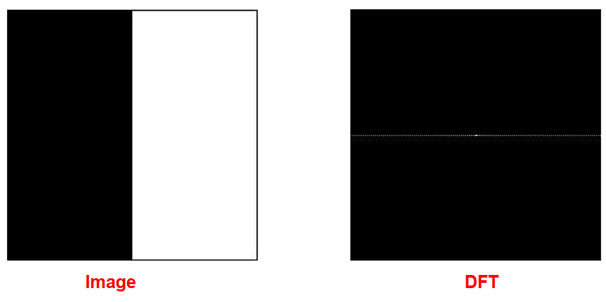
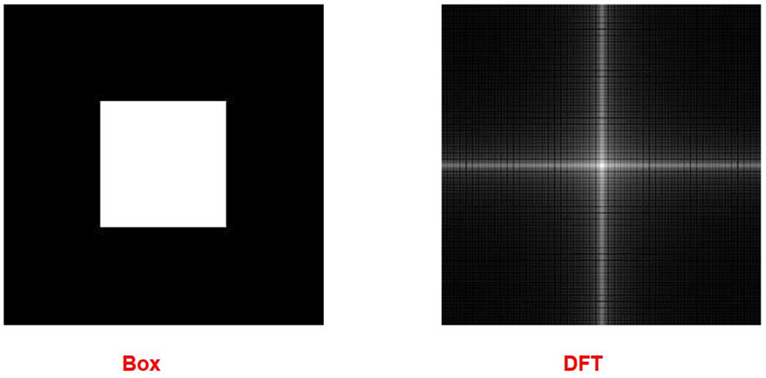
Spectral Techniques (频谱技术) 简介到此结束。这部分的核心是理解Convolution Theorem (卷积定理)，以及\*\*low-pass (低通) $\Leftrightarrow$ smoothing (平滑)\*\* 和 **high-pass (高通) $\Leftrightarrow$ sharpening (锐化)** 之间的对应关系。

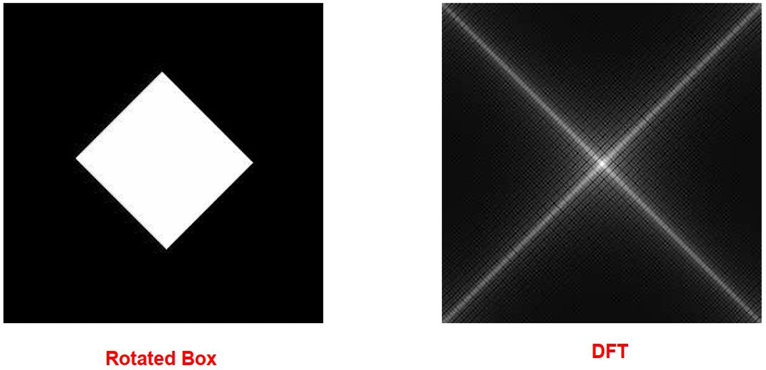
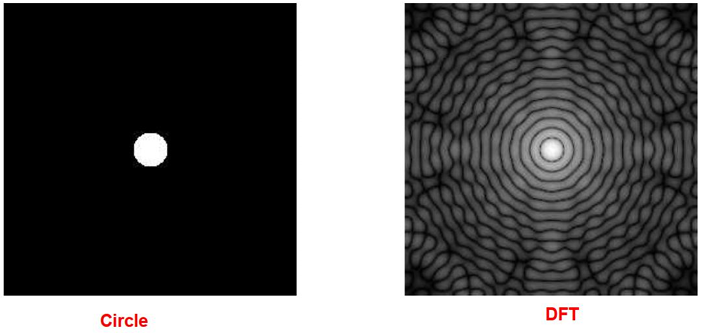
***傅里叶变换对应的 high-level 考点：匹配题***

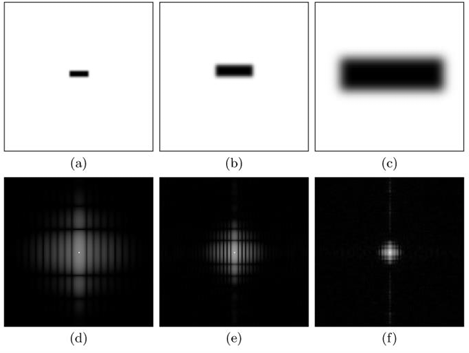
Fourier Transform 前后（空间域 ↔ 频域）统计关系总表：

| **空间域** | **频域** | **解释** |
| --- | --- | --- |
| **结构越窄、越尖锐** | 频谱越宽（高频大量出现） | 窄代表“变化快”→ 高频多 |
| **结构越宽、越模糊** | 频谱越窄（集中在低频） | 宽是“变化慢”→ 高频少 |

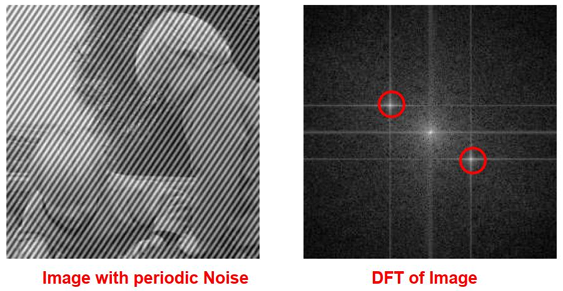
| **空间域中的边或条纹方向** | **频域亮线的方向** |
| --- | --- |
| **竖直边缘** | **水平亮线** |
| **斜 45° 边缘** | **斜 45° 亮线，但方向与边缘垂直** |
| **条纹方向** | **亮点连线方向垂直于条纹** |

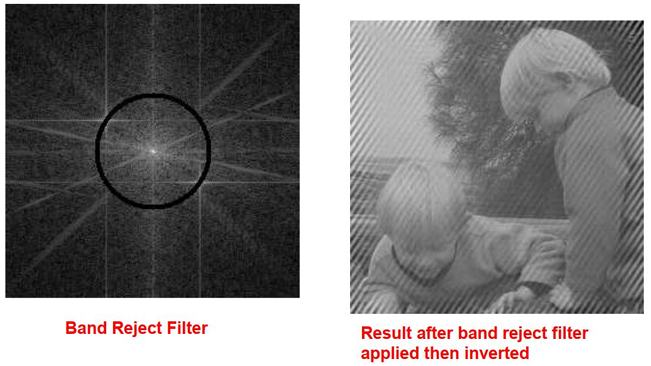
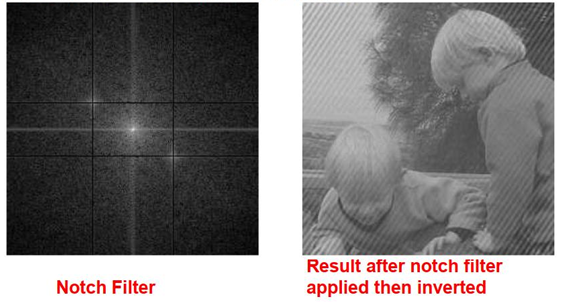
 

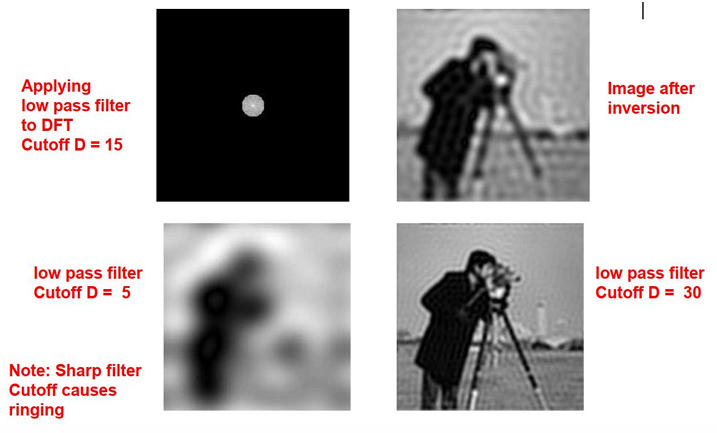
空间域和频率域的大小成反比

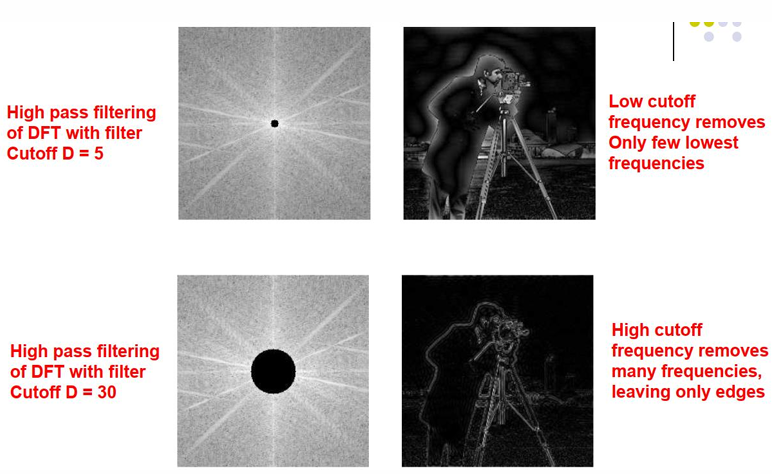
| **空间域** | **频域** | **原因** |
| --- | --- | --- |
| **周期性噪声 / 条纹（sinusoid）** | **两个亮点（±f）** | 正弦波 FT = 两个Dirac δ 点 |
| **二维周期图案（棋盘）** | 网格状多个亮点 | 多重频率叠加 |

去条纹就是遮住亮点



中间是低频+低通滤波，所以中间区域越小，低通越强，越平滑模糊。















不管是开运算还是闭运算，**第二步的目标都是恢复图像**：开对外，闭对内

因此，先腐蚀后膨胀的开运算重点是腐蚀：eliminate thin protrusions (消除细小突起) 和 break narrow bridges (断开窄桥)。

先膨胀后腐蚀的闭运算的重点是膨胀：eliminate small holes (消除小孔洞) 和 fill narrow gaps (填充窄缝)。

