

智能系统设计—Part2—数字图像处理

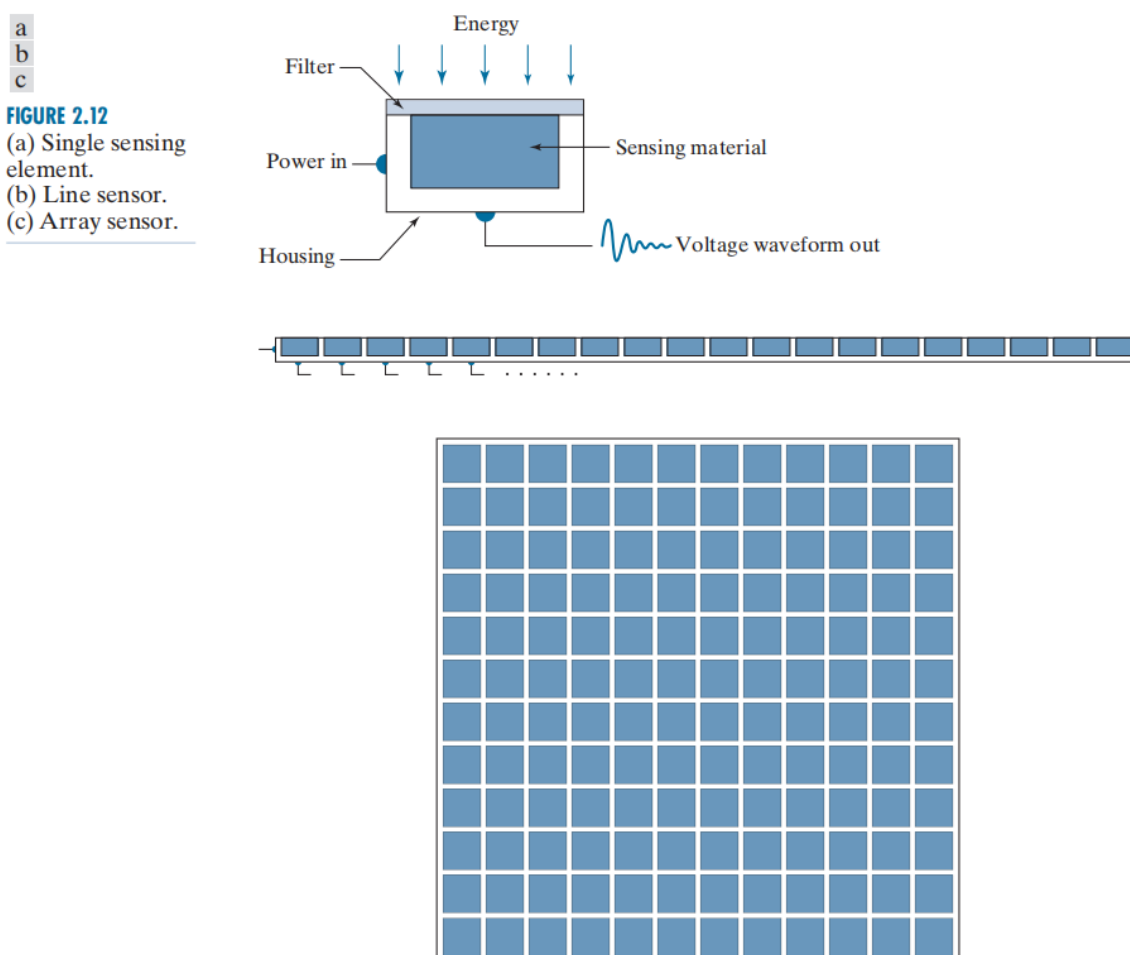
绪论（数字图像处理技术领域）

下面罗列成像技术：

1. 伽马射线成像
2. X射线成像
3. 紫外波段成像
4. 可见光和红外波段成像
5. 无线电波段成像

数字图像基础

图像感知与获取



如图所示传感器可以将照射能量转换为数字图像。原理很简单：组合输入电能和传感器对正被检测能量类型进行响应，将入射能量转换为电压。输出电压波形是传感器的响应，将传感器响应数字化，得到一个数字量。

使用阵列传感器获取图像

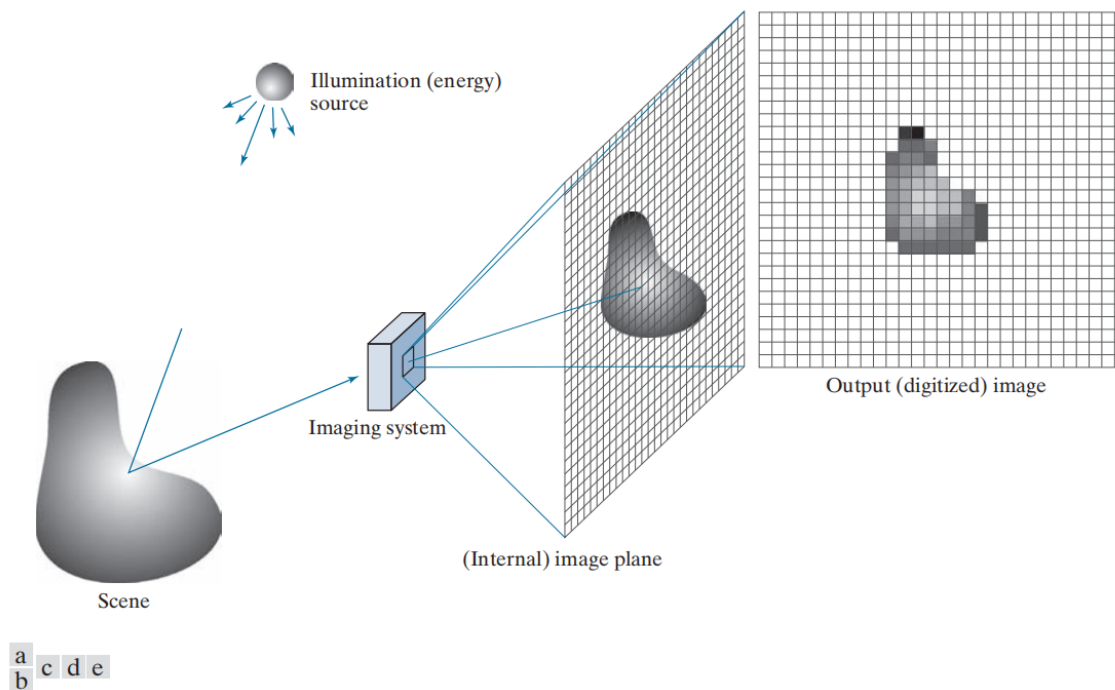
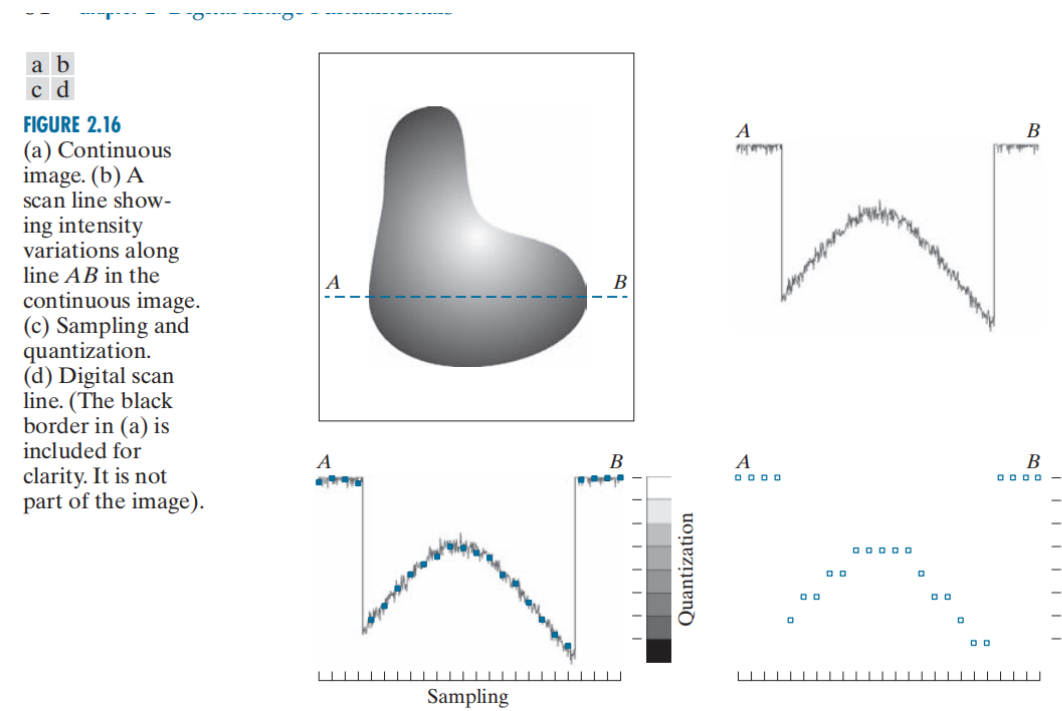


FIGURE 2.15 An example of digital image acquisition. (a) Illumination (energy) source. (b) A scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

如果照射的是光，那么成像系统的前端是一个光学透镜，这个透镜将观察的场景投射到透镜的聚平面上。传感器阵列产生与每个传感器接收到的总光量程正比的输出。数字和模拟电路扫描这些输出并把它们转换成模拟信号，然后由成像系统的其他部分数字化。最终输出是一幅数字图像。

图像取样和量化

基本概念



- 1. **连续图像**：图(a)展示了一个连续的二维图像，其中每一点都有一个光强度值（灰度值）。这是未经过数字化处理的图像。
- 2. **扫描线**：图(b)显示了一条扫描线 AB，该线上的强度值随位置变化。扫描线的强度分布表示了连续图像中沿这条线的灰度变化。

3. **采样 (Sampling)**: 图(c)展示了对扫描线的采样过程。在采样过程中, 沿扫描线 AB 等间隔选取样本点, 这些点的坐标值被记录。采样的结果是将连续的空间坐标离散化, 形成一系列的点。
4. **量化 (Quantization)**: 图(d)进一步对采样点的强度值进行量化。量化是将连续的强度值分为有限的离散级别 (例如8个级别, 从黑到白)。每个采样点的强度值被分配到最接近的量化级别, 从而将连续的强度信息转换为离散的数值。

综上, 采样和量化的定义分别为:

1. **采样**: 采样是指对图像的**空间坐标**进行离散化的过程。
具体来说, 采样将连续图像的空间域 (如 x 和 y 方向的连续坐标) 划分成离散的网格, 每个网格对应一个像素点, 从而使图像能够表示为由像素组成的二维矩阵。
2. **量化**: 量化是指对图像的**像素值 (灰度值或颜色值)**进行离散化的过程。
由于像素值在模拟图像中是连续的, 需要将这些值映射到有限数量的离散值 (即数字级别), 以便用数字形式表示。

数字图像表示

数字图像由许多**像素**组成, 每个像素是图像的最小单元。像素的位置由其在二维网格中的坐标 (x, y) 表示。像素的值 (通常称为**灰度值**或**颜色值**) 表示图像该点的光强度或颜色信息。

灰度值: 对于灰度图像, 每个像素的值是一个表示亮度的单个数值。通常, 灰度值为 **0 到 255** (8位表示), 0表示完全黑色, 255表示完全白色。

对于一个 $M \times N$ 大小的图片来说, 定义离散灰度级数 L 。对于 L 的取值, 通常取2的整数幂次

$$L = 2^k$$

假设离散灰度级是等间隔的, 且它们的区间是 $[0, L - 1]$ 内的整数

有时, 灰度跨越的值域称为动态范围。在这里, 我们将图像系统动态范围定义为系统中最大可度量灰度与最小可检测灰度之比。通常, 上限取决于**饱和度**, 下限取决于**噪声**。**饱和度**表示像素强度达到最大值, 导致区域失去细节; **噪声**是指图像中的随机变化或干扰, 表现为颗粒纹理, 影响图像清晰度。

彩色图像

彩色图像通常使用 **RGB 模型**表示, 每个像素由三个颜色分量组成: **红 (R)**、**绿 (G)**、**蓝 (B)**。

每个分量通常用 8 位表示, 范围是 0 到 255, 因此一个像素总共需要 24 位 (3×8 位)。

- 例如, $(255, 0, 0)$ 表示纯红色, $(0, 255, 0)$ 表示纯绿色, $(0, 0, 255)$ 表示纯蓝色。

空间分辨率(Spatial resolution)

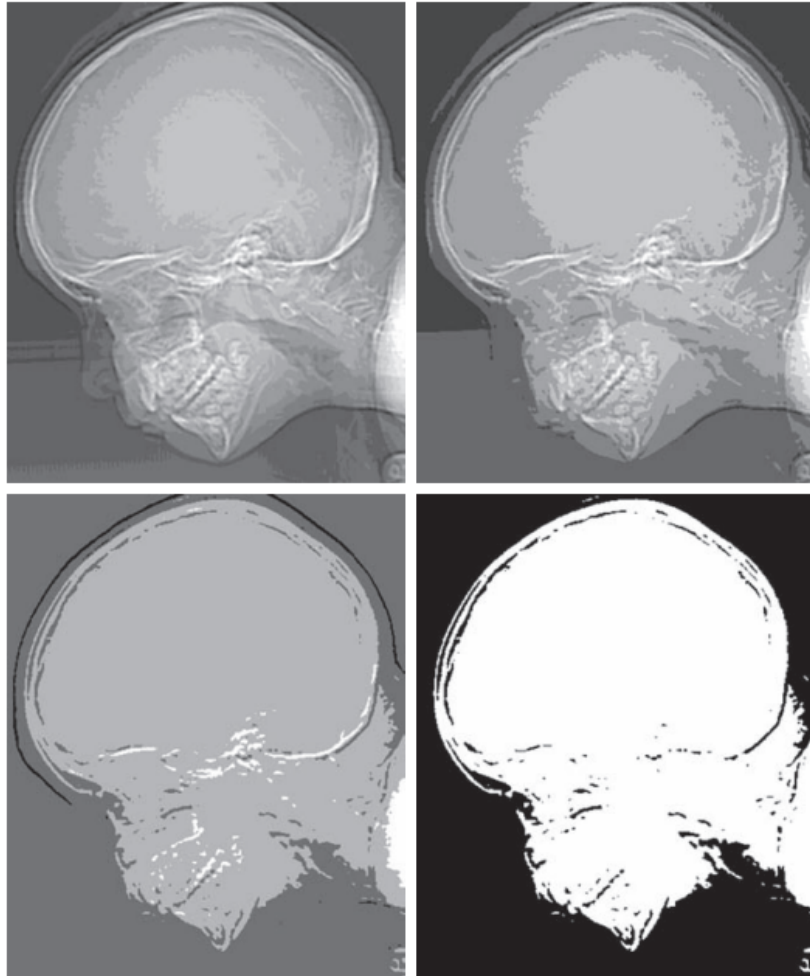
分辨率是指图像中单位长度内包含的像素数量, 通常用 **DPI (dots per inch, 每英寸像素点数)** 来表示。分辨率越高, 图像中的细节就越丰富, 越清晰。

灰度分辨率(Intensity resolution)

灰度分辨率是指在灰度级中可分辨的最小变化。灰度分辨率通常是指量化灰度级时所用的比特数。例如, 对于一个被量化为256级的图像, 最常用的数是8bit。

e	f
g	h

FIGURE 2.24
(Continued)
(e)-(h) Image displayed in 16, 8, 4, and 2 intensity levels.

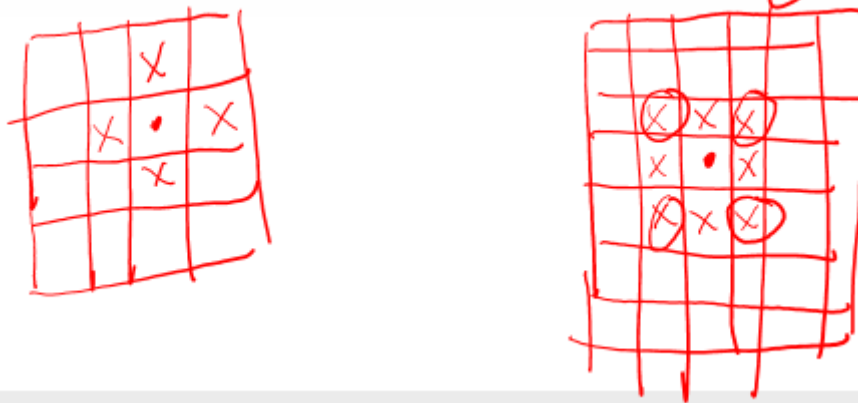


图中分别展示了灰度级为16,8,4,2的图像。如果想把一个高分辨率的图像转换为低分辨率的图像，例如将256级转换为128级灰度。

$$I_{128} = \left\lfloor \frac{I_{256}}{2} \right\rfloor \times 2$$

像素的相邻像素

not form a closed path, but its outer boundary does.



左图和右图分别展示了4邻域和8邻域。如果一个像素从一点移动到另一点，4邻域只能上下左右移动，而8邻域更加灵活，可以斜对角线(diagonal)移动

数字图像处理所用的基本数学工具

使用图像相加（平均）降低噪声

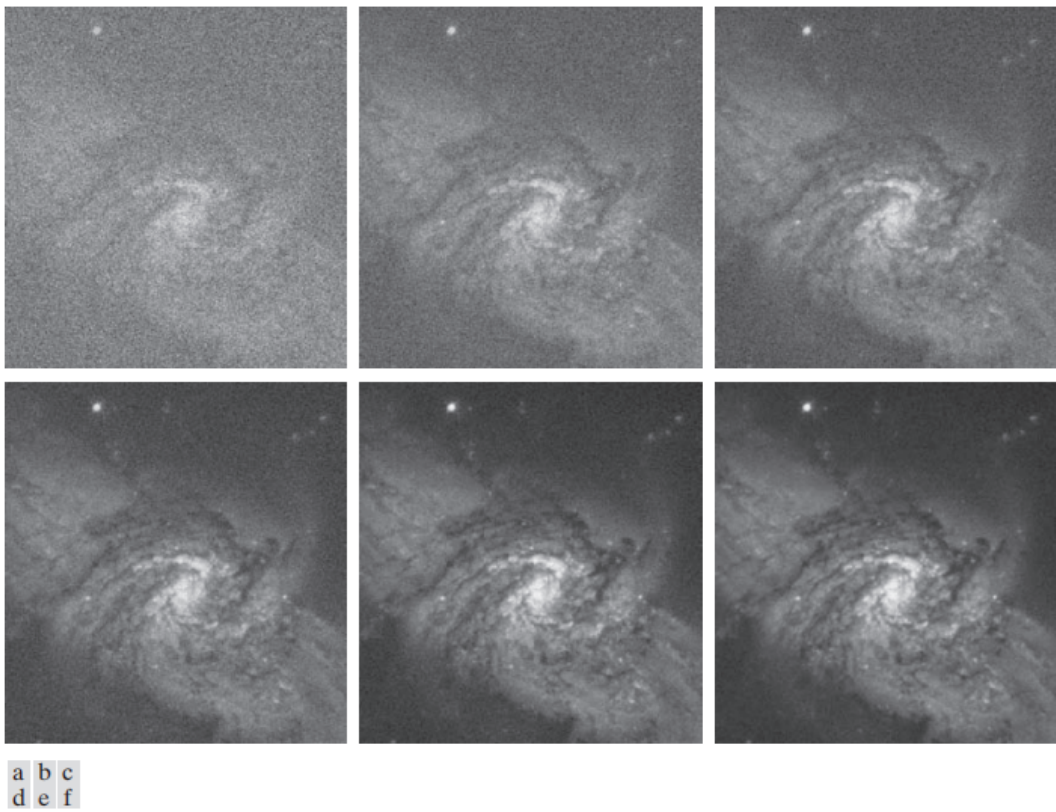
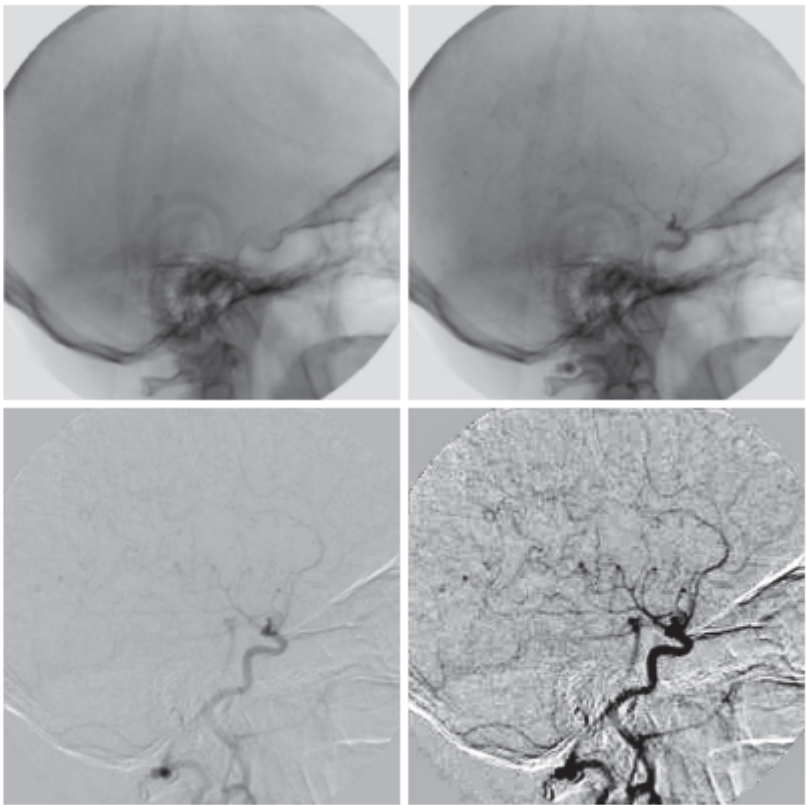


FIGURE 2.29 (a) Image of Galaxy Pair NGC 3314 corrupted by additive Gaussian noise. (b)-(f) Result of averaging 5, 10, 20, 50, and 1,00 noisy images, respectively. All images are of size 566×598 pixels, and all were scaled so that their intensities would span the full $[0, 255]$ intensity scale. (Original image courtesy of NASA.)

使用图像相减增强图像之间的差

a b
c d

FIGURE 2.32 Digital subtraction angiography. (a) Mask image. (b) A live image. (c) Difference between (a) and (b). (d) Enhanced difference image. (Figures (a) and (b) courtesy of the Image Sciences Institute, University Medical Center, Utrecht, The Netherlands.)



如第四张图所示，这是增强的差值

集合运算和逻辑运算

交集 (Intersection, \cap)：用于提取两个图像中共同的部分。

并集 (Union, \cup)：用于合并两个图像中所有的像素。

差集 (Difference, $-$)：用于获取一个图像中属于A但不属于B的像素。

补集 (Complement, \sim)：将像素值取反，即前景变成背景，背景变成前景。

邻域运算(Neighborhood Operations)

邻域运算基于像素所在的局部区域（邻域）的像素值进行处理。例如，可以计算邻域内像素的平均值、最大值、最小值等。

$$g(x, y) = \frac{1}{mn} \sum_{(r, c) \in S_{xy}} f(r, c)$$

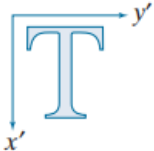
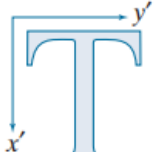
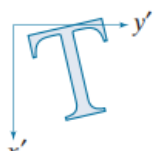
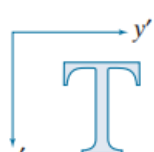
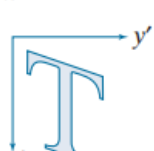
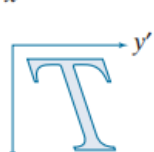
邻域运算的主要作用：

- 图像平滑（去噪）**：在例子中，邻域运算使用的是一个大小为 41×41 的矩形邻域来计算像素的平均值，这是一种平滑滤波的操作。它通过对每个像素邻域内的值取平均，有效地去除了图像中的随机噪声。
- 增强局部特征**：邻域运算可以用于增强图像中的某些局部特征，例如边缘检测（通过梯度或差分计算）或纹理提取。
- 图像的像素级变换**：邻域运算会对每个像素重复上述操作，通过移动邻域窗口，遍历图像所有像素并计算输出图像的每个值。

几何变换(Geometric Transformations)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

其中 (x, y) 是原图像中像素的坐标， (x', y') 是变换后图像中像素的坐标。这种坐标变换称为 *仿射映射 (affine transformations)*，它可以实现缩放、旋转、平移或剪切变换。

Transformation Name	Affine Matrix, A	Coordinate Equations	Example
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x$ $y' = y$	
Scaling/Reflection (For reflection, set one scaling factor to -1 and the other to 0)	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = c_x x$ $y' = c_y y$	
Rotation (about the origin)	$\begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x \cos \theta - y \sin \theta$ $y' = x \sin \theta + y \cos \theta$	
Translation	$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x + t_x$ $y' = y + t_y$	
Shear (vertical)	$\begin{bmatrix} 1 & s_v & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x + s_v y$ $y' = y$	
Shear (horizontal)	$\begin{bmatrix} 1 & 0 & 0 \\ s_h & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x$ $y' = s_h x + y$	

图像变换

$$T(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) r(x, y, u, v)$$

其中, $f(x, y)$ 是输入图像, $r(x, y, u, v)$ 被称为**正变换核** (forward transformation kernel) , 并且公式(2-55)针对 $u = 0, 1, 2, \dots, M - 1$ 和 $v = 0, 1, 2, \dots, N - 1$ 计算。与之前一样, x 和 y 是空间变量, 而 M 和 N 是 f 的行和列维度。变量 u 和 v 被称为**变换变量** (transform variables) 。
 $T(u, v)$ 被称为 $f(x, y)$ 的**正变换** (forward transform) 。给定 $T(u, v)$, 我们可以通过 $T(u, v)$ 的**反变换** (inverse transform) 恢复 $f(x, y)$ 。

变换的性质取决于变换核。在数字图像处理中, 一种特别重要的变换是傅里叶变换。

图像压缩(Image Compression)

数据冗余(data redundancies)

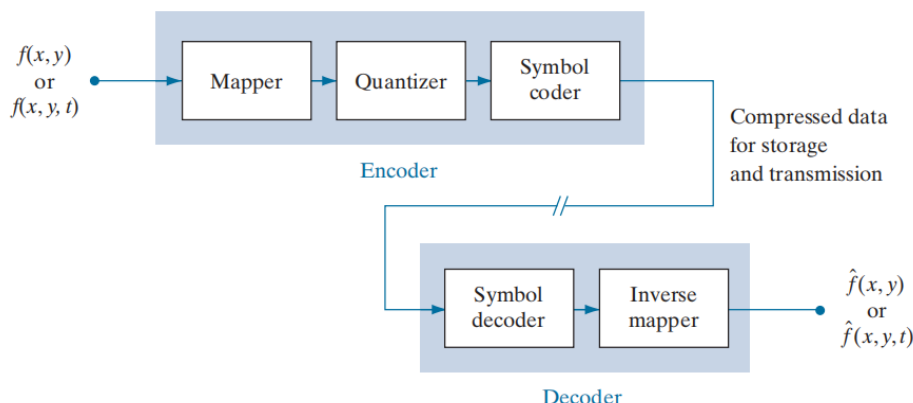
- 编码冗余(Coding redundancy):** 信息使用编码符号 (如字母、数字、二进制等) 来表示。当用这些符号组合来传递信息时, 经常会出现冗余。例如, 8位的二进制编码通常包含比表示实际强度所需更多的比特。换句话说, **可以用更少的符号来传递同样的信息, 从而减少冗余。**
- 空间和时间冗余(Spatial and temporal redundancy):** 在图像或视频中, 相邻像素之间往往具有很强的相似性或依赖性。这种空间上的重复导致信息不必要地被重复存储或传递。在视频中, 相邻帧之间的像素也经常非常相似, 形成时间上的冗余。

3. **无关信息(Irrelevant information)**: 图像中存在一些信息, 对于人类视觉系统来说并不重要, 或者和图像的主要用途无关。这些信息对实际使用没有帮助, 因此是冗余的。

图像压缩模型

FIGURE 8.5

Functional block diagram of a general image compression system.

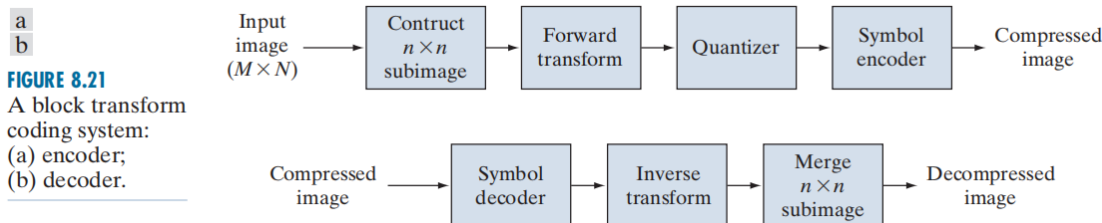


编码器 (Encoder)

1. **映射器**: 将输入的原始图像数据 $f(x, y)$ 或视频数据 $f(x, y, t)$ 转换成一种适合压缩的形式。提取出图像或视频数据的核心特征, 去掉一些冗余信息。各种变换在这一步进行, 例如傅里叶变换。
2. **量化器**: 进一步压缩数据, 通过用有限的级别 (如整数值) 来表示连续的数值, 减少数据量。
3. **符号编码器**: 将数据转换为适合存储和传输的符号形式 (比如二进制代码)。霍夫曼编码在这一步执行。

解码器(Decoder)

1. **符号解码器**: 将压缩的数据流解码为量化后的数据。
2. **反映射器**: 将量化后的数据还原成接近原始图像或视频的形式 $\hat{f}(x, y)$ 或 $\hat{f}(x, y, t)$ 。



该图展示了一个JPEG格式图像压缩和解压缩的过程。第一步通过**构建 $n \times n$ 的多幅子图像**, 可以减少复杂度, 同时便于后续的变换操作。然后对其进行**前向变换**, 将图像从空间域转化为频域, 将图像中对视觉影响较小的高频信息分离出来, 便于后续的压缩。**量化器**通过舍弃不重要的高频信息 (对图像质量影响不大) 来减少数据量, 但可能导致部分数据丢失。这里的量化阶段选择性的删除携带信息最少得系数, 因为这些系数对重建子图像的质量影响最小。最后**符号编码器**通过编码方法 (例如霍夫曼编码或算术编码) 进一步压缩数据, 生成最终的压缩图像。

编码冗余

假设用区间 $[0, L - 1]$ 内一个离散随机变量 r_k 来表示一幅 $M \times N$ 图像的灰度, 且每个 r_k 出现的概率为 $p_r(r_k)$, 则:

$$p_r(r_k) = \frac{n_k}{MN}, \quad k = 0, 1, 2, \dots, L - 1$$

其中, L 是灰度值的数量, n_k 是第 k 级灰度在图像中出现的次数。若用于表示每个 r_k 值的比特数为 $l(r_k)$, 则表示每个像素所需的平均比特数为:

$$L_{avg} = \sum_{k=0}^{L-1} l(r_k) p_r(r_k)$$

TABLE 8.1
Example of
variable-length
coding.

r_k	$p_r(r_k)$	Code 1	$l_1(r_k)$	Code 2	$l_2(r_k)$
$r_{87} = 87$	0.25	01010111	8	01	2
$r_{128} = 128$	0.47	01010111	8	1	1
$r_{186} = 186$	0.25	01010111	8	000	3
$r_{255} = 255$	0.03	01010111	8	001	3
r_k for $k \neq 87, 128, 186, 255$	0	—	8	—	0

如图的编码2，这种方式的思想为对图像中更大可能出现的灰度值分配较少的比特，而对更小可能出现的灰度值分配较多的比特。

霍夫曼编码(Huffman Coding)

霍夫曼编码 (Huffman Coding) 是一种**无损数据压缩算法**，由 David A. Huffman 在 1952 年提出。霍夫曼编码是一种**基于符号出现概率的最优编码方法**，目标是为概率更高的符号分配较短的二进制码，为概率较低的符号分配较长的二进制码，从而实现数据压缩。

霍夫曼编码分为两步，信源简化和编码的分配

信源简化

FIGURE 8.7
Huffman source
reductions.

Original source		Source reduction			
Symbol	Probability	1	2	3	4
a_2	0.4	0.4	0.4	0.4	0.6
a_6	0.3	0.3	0.3	0.3	0.4
a_1	0.1	0.1	0.2	0.3	
a_4	0.1	0.1	0.1		
a_3	0.06	0.1			
a_5	0.04				

从最低概率的符号开始，两两合并，生成新的节点。

- 第 1 步：合并 a_5 (0.04) 和 a_3 (0.06)，生成概率为 0.1 的新节点。
- 第 2 步：合并新的 0.1 节点和原始的 a_4 (0.1)，生成新的 0.2 节点。
- 第 3 步：合并 0.2 和 a_6 (0.3)，生成新的 0.5 节点。
- 第 4 步：合并 0.4 和 0.5，生成最终的根节点 (1.0)。

合并过程本质上是在构建一棵二叉树，最终的根节点概率为 1.0，表示所有符号都包含在这棵树中。

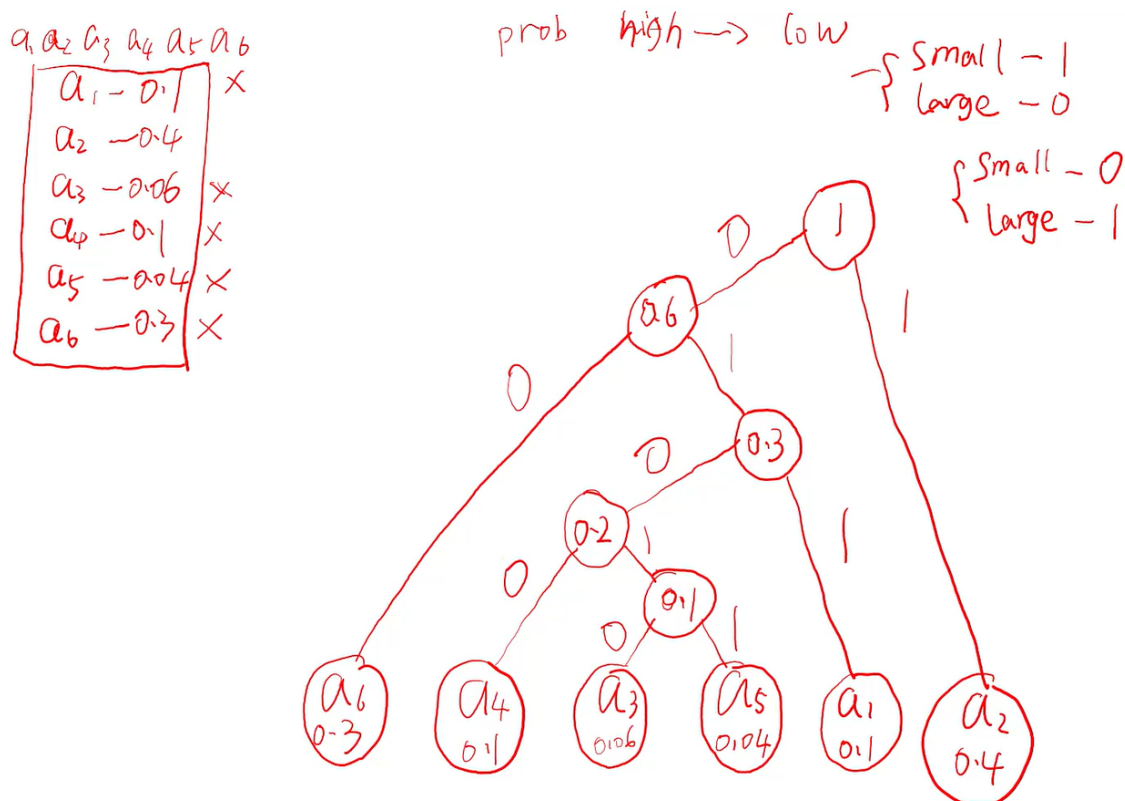
霍夫曼编码分配过程

FIGURE 8.8
Huffman code
assignment
procedure.

Original source			Source reduction			
Symbol	Probability	Code	1	2	3	4
a_2	0.4	1	0.4	1	0.4	1
a_6	0.3	00	0.3	00	0.3	00
a_1	0.1	011	0.1	011	0.2	010
a_4	0.1	0100	0.1	0100	0.1	011
a_3	0.06	01010	0.1	0101		
a_5	0.04	01011				

符号 a_2 的概率最大 (0.4)，所以分配了最短的编码 00。

符号 a_5 的概率最小 (0.04) , 分配了最长的编码 01011。



可以使用这种方法构建霍夫曼树, 这里定义左枝为0, 右枝为1。

图像压缩中误差的衡量方法

在图像压缩中, 原始图像 $f(x, y)$ 和压缩后重建的图像 $\hat{f}(x, y)$ 可能存在差异。为了评估这种差异的大小, 通常使用**均方误差 (MSE)** 和 **均方根误差 (RMSE)**。

$$MSE = \frac{1}{M \cdot N} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x, y) - \hat{f}(x, y)]^2$$
$$RMSE = \sqrt{MSE}$$

其中:

- M 和 N 是图像的宽度和高度
- $f(x, y)$ 是原始图像坐标在 (x, y) 的像素值
- $\hat{f}(x, y)$ 是压缩后重建图像在同一坐标 (x, y) 的像素值

块变换编码(BLOCK TRANSFORM CODING)

变化的选择

离散余弦变换 (DCT) 和 卡洛能-洛伊夫变换 (KLT) 是两种不同的编码技术。下面对比这两种技术

信息压缩能力 (Information Packing Ability)

KLT:

- 理论上, KLT 是最佳的, 因为它能够为每幅图像动态生成基础变换, 最大化地将信息集中到少数的系数中。
- 对于任何输入图像和任何数量的保留系数, KLT 都能**最小化均方误差 (MSE)**, 因此它在信息打包效率上优于 DCT。

DCT:

- 虽然 DCT 的信息压缩能力不如 KLT，但它仍然接近最优（非常接近 KLT）。
- 因为 DCT 是基于固定变换矩阵的，因此无法像 KLT 那样完全适应每幅图像。

计算复杂度

KLT:

- KLT 是图像依赖的，需要根据每个输入图像的统计特性动态计算变换矩阵（即基础图像）。
- 由于这种依赖性，KLT 的计算复杂度高，实现困难，尤其是在实际应用中处理子块图像时。

DCT:

- DCT 使用固定的变换矩阵，与输入图像无关（图像独立）。
- 因为固定变换矩阵，DCT 的实现简单，计算效率高，更适合硬件加速和实时应用。

实际应用场景中的优势

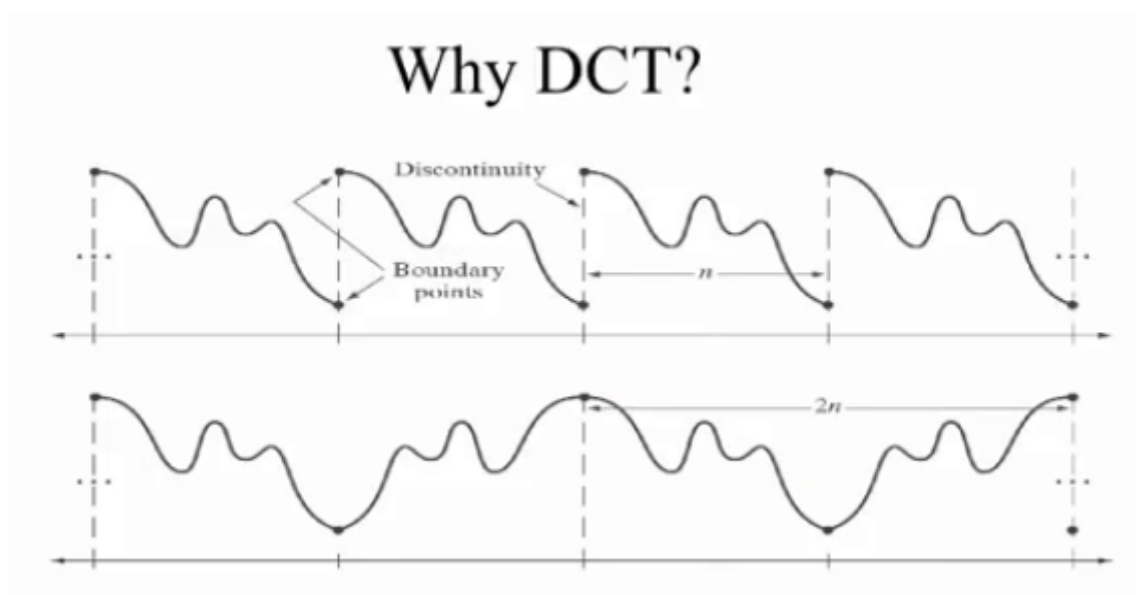
KLT:

- 由于计算复杂度高，KLT 在实际应用中很少使用，更多是一种理论上的最优方案。

DCT:

- DCT 在信息打包能力和计算复杂度之间提供了良好的折中，因此成为了实际中最常用的变换方法。

与傅里叶变换相比，DCT的优势



上方的图是傅里叶变换，下图是DCT变换。这是时域图

- **傅里叶变换:**
 - 每个图像块（例如 n 长度）会在边界点直接重复。
 - 这种重复可能导致**边界不连续性 (Discontinuity)**，因为块之间的值通常不匹配。
 - 这种边界不连续性会引发图像压缩中的**伪影 (blocking artifacts)**，导致压缩图像在分块边缘出现可见的瑕疵。
- **离散余弦变换 (DCT) :** 假设信号在边界处是对称镜像的
 - 每个图像块以对称的方式延拓形成周期。
 - 这种对称延拓使得信号在边界处保持连续性，避免了块之间的值差异，从而显著减少了边界伪影的出现。

综上，傅里叶变换的直接周期性，可能造成**边界处的不连续**；而离散余弦变换的对称周期性，能够**消除边界处的不连续性**。在真实的图像中，可以观察到使用DCT变换压缩后的图像更加的平滑。

图像增强和过滤(Image Enhancement and Filtering)

空间域中一个基本表达式：

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

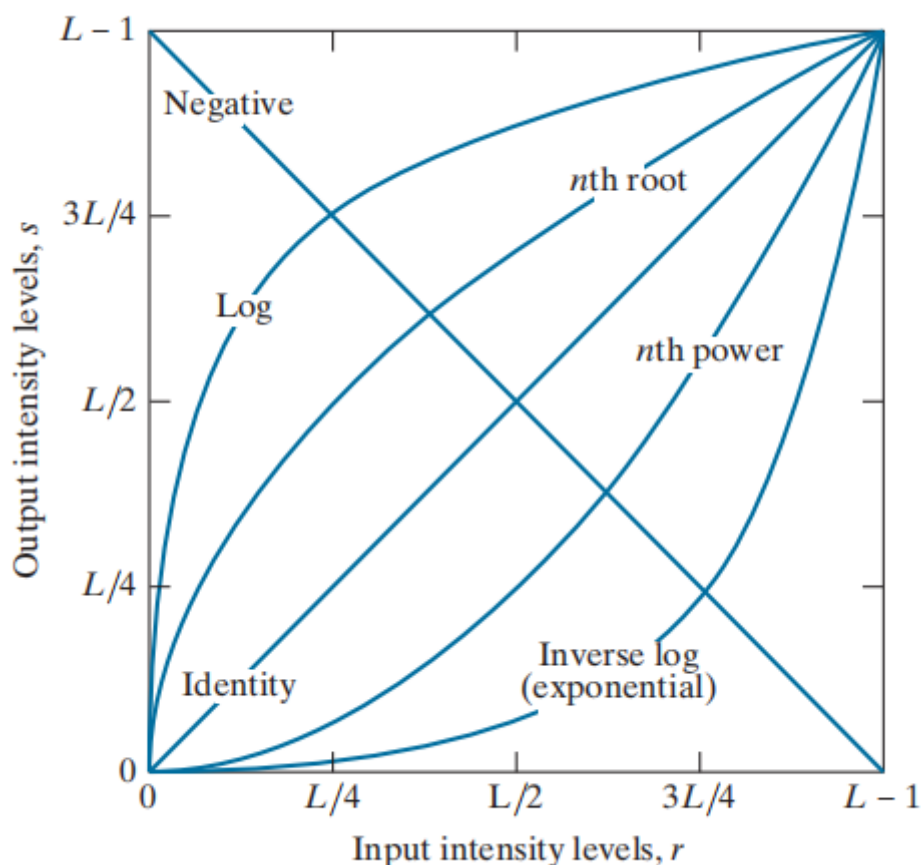
其中 $f(x, y)$ 是输入图像， $g(x, y)$ 是输出图像， T 是在点 (x, y) 的一个邻域上定义的针对 f 的算子。本章中， T 也称为灰度变换函数(intensity transformation function)。

一些基本的灰度变换函数

图像反转

灰度级在区间 $[0, L - 1]$ 内的反转图像的形式为：

$$s = L - 1 - r$$



伽马变换

$$s = cr^\gamma$$

分段线性函数

1. **对比度拉伸**
2. **灰度级分层**：目的是突出图像中特定灰度区间

直方图处理

设 r_k , 其中 $k = 0, 1, 2, \dots, L-1$, 表示 L 级数字图像 $f(x, y)$ 的灰度级。 f 的**未归一化直方图**定义为:

$$h(r_k) = n_k \quad \text{对于 } k = 0, 1, 2, \dots, L-1$$

其中, n_k 表示灰度值为 r_k 的**像素数量**, 灰度值的划分被称为**直方图区间** (histogram bins)。类似地, f 的**归一化直方图(normalized histogram)**定义为:

$$p(r_k) = \frac{h(r_k)}{MN} = \frac{n_k}{MN}$$

其中, M 和 N 分别是图像的行数和列数。

直方图均衡化

图像处理中一个特别重要的变换函数是

$$s = T(r) = (L-1) \int_0^r p_r(w) dw$$
$$\frac{ds}{dr} = \frac{dT(r)}{dr} = (L-1) \frac{d}{dr} \left[\int_0^r p_r(w) dw \right] = (L-1) p_r(r)$$
$$p_s(s) = p_r(r) \left| \frac{dr}{ds} \right| = p_r(r) \left| \frac{1}{(L-1) p_r(r)} \right| = \frac{1}{L-1} \quad 0 \leq s \leq L-1$$

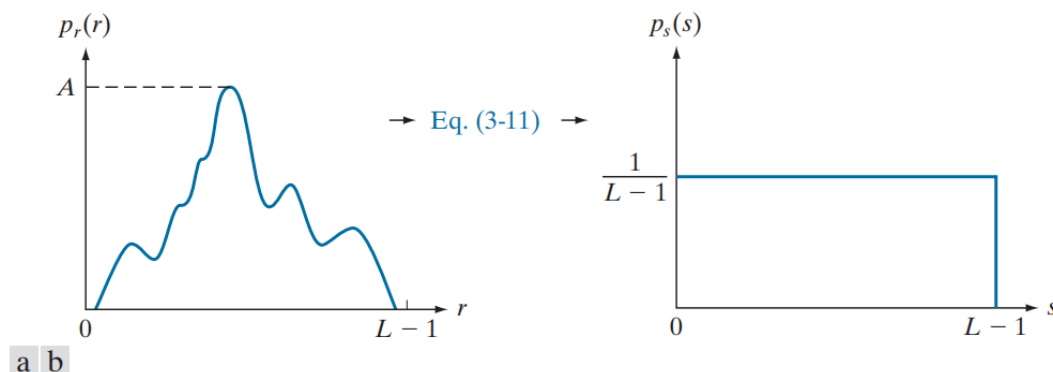


FIGURE 3.18 (a) An arbitrary PDF. (b) Result of applying Eq. (3-11) to the input PDF. The resulting PDF is always uniform, independently of the shape of the input.

变换函数的离散形式为:

$$s_k = T(r_k) = (L-1) \sum_{j=0}^k p_r(r_j) \quad k = 0, 1, 2, \dots, L-1$$

直方图的规定化 (Histogram Specification)

直方图的规定化 (也称为直方图匹配) 是图像处理中的一种技术, 用来将图像的灰度分布调整为目标分布形状。与直方图均衡化不同的是, 直方图规定化可以将一幅图像调整为任意的灰度分布, 而不仅仅是均匀分布。

总结

直方图规定化通过以下步骤实现:

1. **原始直方图均衡化**, 得到均衡化灰度值 s_k 。

2. 计算目标直方图的累积分布函数 (CDF)，得到目标灰度级 $G(z_q)$ 。
3. 建立映射关系，将 s_k 值映射到最接近的目标灰度级 z_q 。
4. 生成规定化图像，将原始图像的灰度级替换为目标分布对应的灰度级。

空间滤波基础(Fundamentals of spatial filtering)

线性空间滤波是图像处理中的一种操作，它的核心思想是利用一个称为“核”或“滤波器”的小矩阵对图像进行处理，从而实现增强或提取某些特定特征。在这个过程中，空间滤波的主要操作是通过“空间相关 (spatial correlation)”或“卷积 (convolution)”计算得出的。

图像复原与重建

