Digital Image Processing

COSC 6380/4393

Lecture – 12

Sept. 28th, 2023

Pranav Mantini

Slides from Dr. Shishir K Shah and Frank (Qingzhong) Liu

Point Operations

POINT OPERATIONS

- A point operation on an image **I** is a **function** f that maps **I** to another image **J** by operating on **individual pixels** in **I**:

  $$J(i, j) = f[I(i, j)],\ 0 <= i, j <= N-1$$

- The same function f is applied at every image coordinate
- This is different from **local operations** such as OPEN, CLOSE, etc., since those are functions of both I(i, j) **and its neighbors**

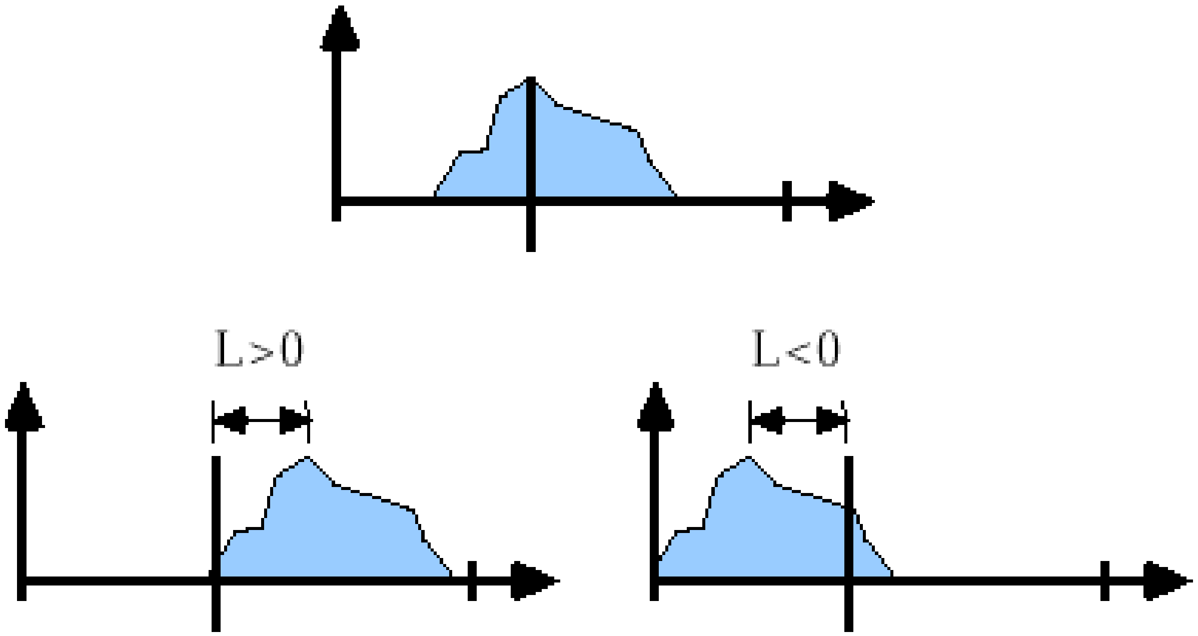
LINEAR POINT OPERATIONS

- **Linear point operations** are the simplest class of point operations

$$F(X) = P.X + L$$

Image Offset

- If $L < 0$, **J** will be a **dimmed** version of the image **I**
- Adding offset L **shifts** the histogram by amount L to left or right:



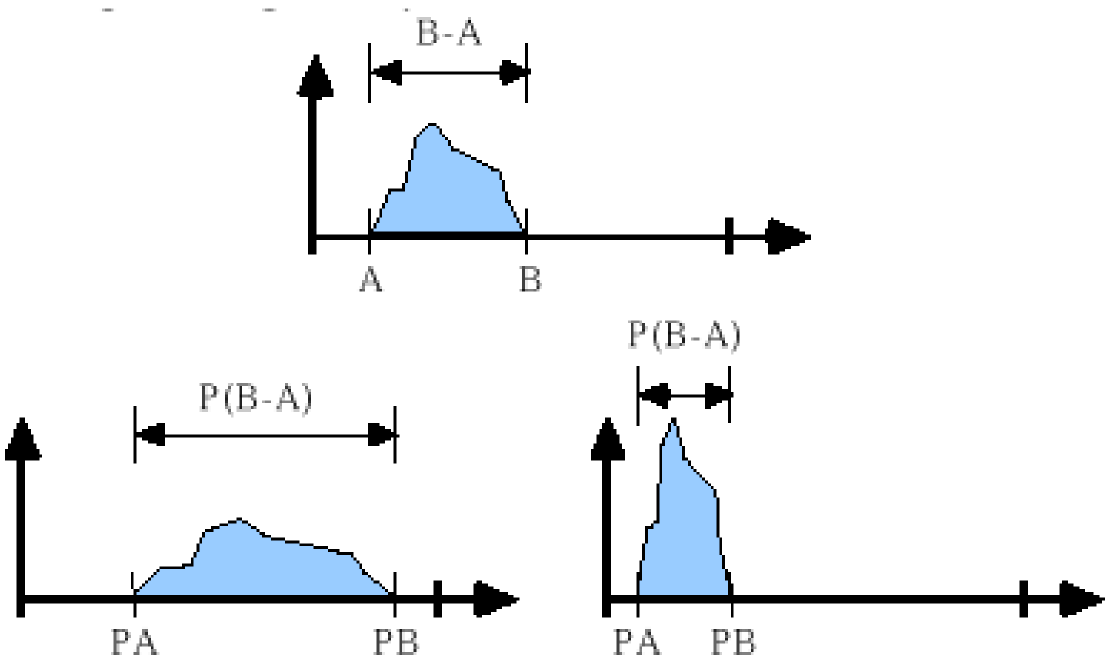
Histograms of additive image offsets

- The input and output histograms are related by:

$$\mathbf{H_J}(k) = \mathbf{H_I}(k-L)$$

Image Scaling

- If P < 1, **J** will have a **narrower grey-level range** than **I**
- Multiplying by a constant P **stretches** or **compresses** the "width” of the image histogram by a factor P:



Linear Point Operations: Offset & Scaling

- Suppose L and P are real numbers (not necessarily integers)
- A **linear point operation** on **I** is defined by the function

  $J(i, j) = P \cdot I(i, j) + L$, for $0 <= i, j <= N-1$

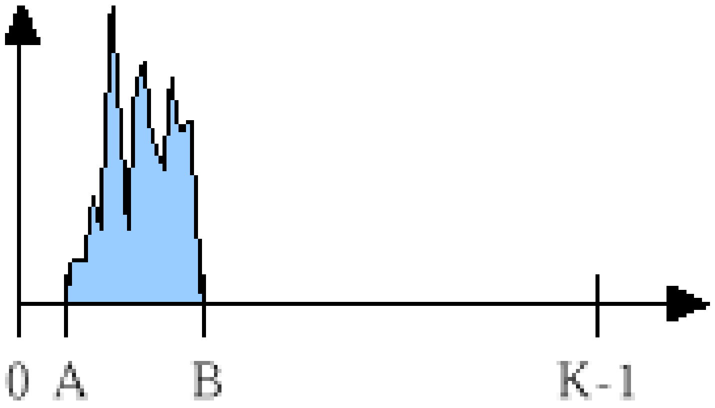
- In practice:

  $J(i, j) = \mathrm{INT}[\, P \cdot I(i, j) + L + 0.5 \,]$, for $0 <= i, j <= N-1$

- The image **J** is a version of **I** that has been scaled and given an additive offset

Full-Scale Contrast Stretch

- The **most common** linear point operation. Suppose **I** has a compressed histogram:



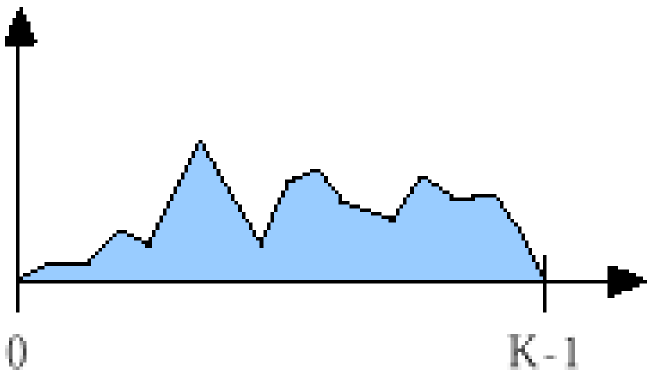
- Let A and B be the min and max gray levels in **I**
- Define

$$J(i, j) = P \cdot I(i, j) + L$$

- such that $P \cdot A + L = 0$ and $P \cdot B + L = (K-1)$

Full-Scale Contrast Stretch

- The result of solving these **2 equations in 2 unknowns** (P, L) is an image **J** with a full-range histogram:



- The solution to the above equations is

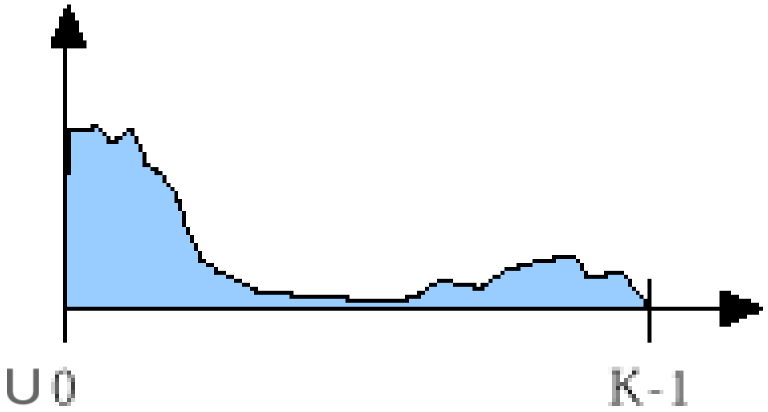
$$P = \left(\frac{K-1}{B-A}\right) \quad \text{and} \quad L = -A\left(\frac{K-1}{B-A}\right)$$

or

$$J(i, j) = \left(\frac{K-1}{B-A}\right)\left[I(i, j) - A\right]$$

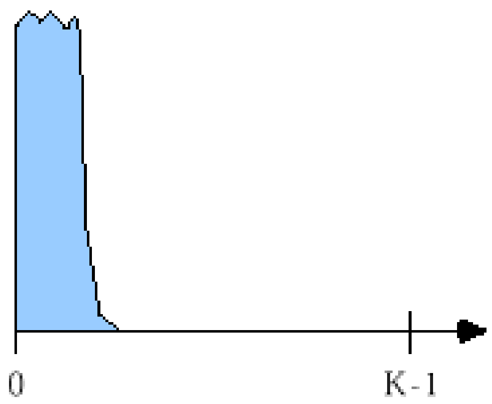
Logarithmic Range Compression

- **Motivation**: An image may contain information-rich, smoothly-changing low intensities - and small very bright regions
- Useful for detecting faint **objects**
- The bright pixels will dominate our visual perception of the image
- A typical histogram:

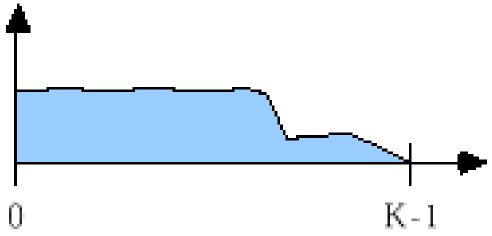


Logarithmic Range Compression

- **Logarithmic transformation** $J(i, j) = \log[1+I(i, j)]$ nonlinearly **compresses** and **equalizes** the gray-scales
- Bright intensities are compressed much more heavily - thus **faint details** emerge
- A full-scale contrast stretch then utilizes the full gray-scale range:



logarithmic transformation



stretched contrast

HISTOGRAM SHAPING

- Apply point operation such that the intensity histogram has a desired shape (target shape)
- Often times, the transformation function is non-linear.
- We now describe methods for **histogram shaping**.
- Accomplished by point operations: object **shape** and **location** are **unchanged**.

DEFINITION

- Define the **normalized histogram**:

$$\mathbf{p_I}(k) = \left(\frac{1}{N^2}\right) \mathbf{H_I}(k)\ ;\ k = 0, \ldots, K\text{-}1$$

- These values **sum to one**: $\sum_{k=0}^{K-1} \mathbf{p_I}(k) = 1$
- Here $\mathbf{p_I}(k)$ is the **probability** that gray-level k will occur (at any given pixel)
- $\mathbf{p_I}(k) \approx$ probability of gray-level k
- The **cumulative histogram** is

$$\mathbf{P_I}(r) = \sum_{k=0}^{r} \mathbf{p_I}(k)\ ;\ r = 0, \ldots, K\text{-}1$$

- $\mathbf{P_I}(r)$ is a nondecreasing function, and $\mathbf{P_I}(K\text{-}1) = 1$.

INTERPRETATION

- With the probabilistic interpretation, at a point (i, j):
- $\mathbf{P_I}(r) = Pr\{I(i, j) \leq r\}$

CONTINUOUS HISTOGRAMS

- Suppose **p**(x) and **P**(x) are **continuous**: they may be regarded as probability density (pdf) and cumulative distribution (cdf).
- $\mathbf{P}^{-1}(x)$ exists or can be defined **by convention**.

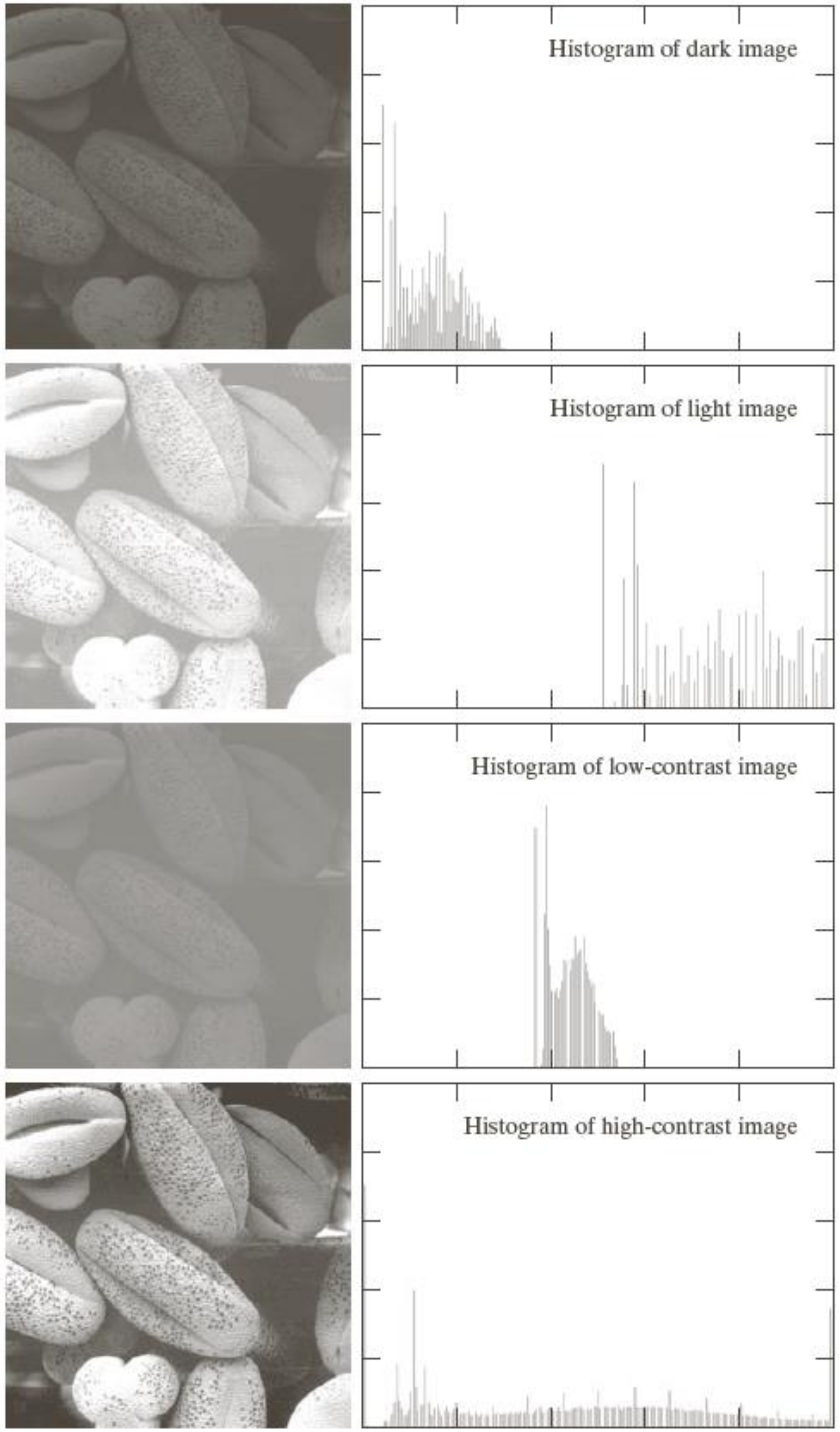
$$If\ Y = F(X),$$
$$F - a\ transformation\ function$$

CDF:

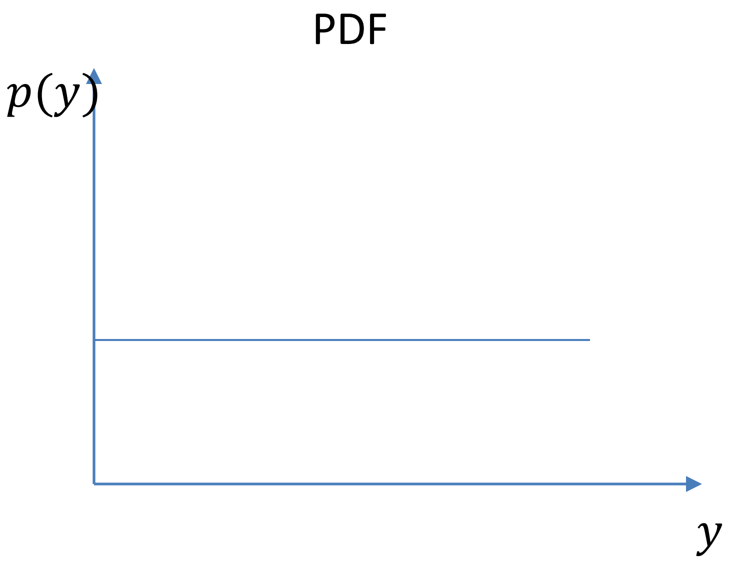
$$\boldsymbol{if\ y = F(x) \Rightarrow y = P_Y{}^{-1}\left(P_X(x)\right)}$$

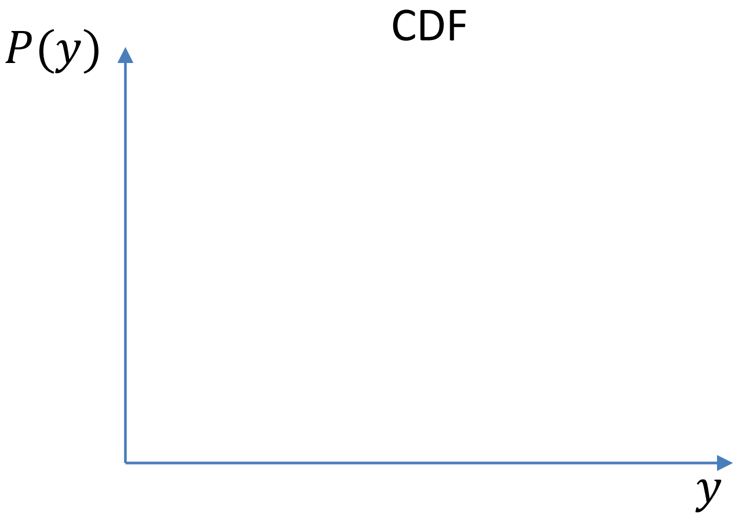
HISTOGRAM SHAPING

- We now describe methods for **histogram shaping**.
- Accomplished by point operations: object **shape** and **location** are **unchanged**.
- **<u>Histogram Flattening (Uniform)</u>**
- An image with a **flat** histogram makes rich use of the available gray-scale range. This might be an image with
  - Smooth gradations in gray scale covering many gray levels
  - Lots of texture covering many gray levels

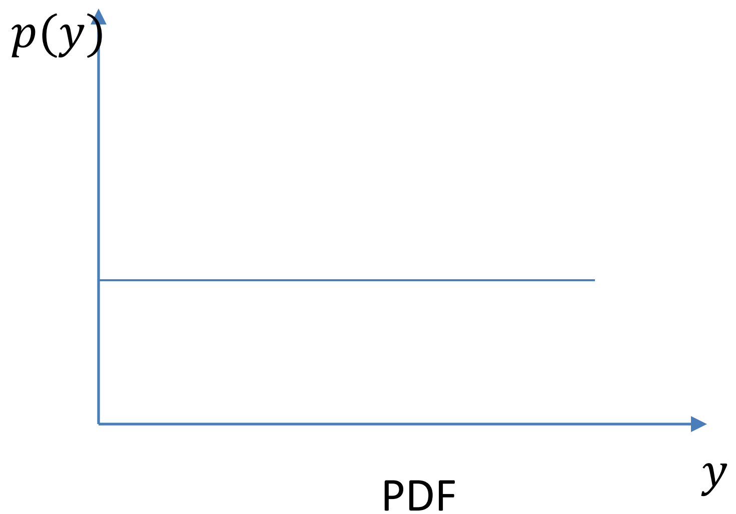

Histogram of dark image
Histogram of light image
Histogram of low-contrast image
Histogram of high-contrast image

Uniform Distribution

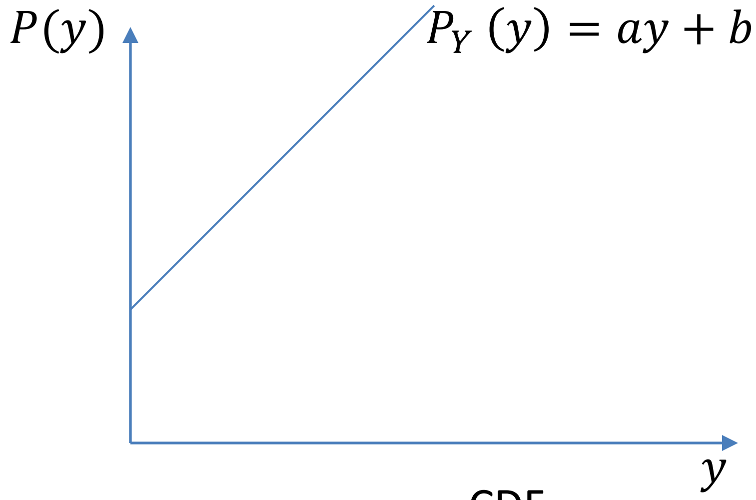




Uniform Distribution



PDF



CDF

$$y \propto P_Y(y)$$

Histogram Flattening

$$if\ y = F(x) \Rightarrow y = {P_Y}^{-1}\big(P_X(x)\big), P_Y(y) = P_X(x)$$
$$P_Y(y) = \mathrm{ay} + \mathrm{b},$$
$$P_X(x) = ay + b \Rightarrow y \propto P_X(x)$$

We can obtain an image **J** with an **approximately** flat histogram from an image **I** by the following procedure.

HISTOGRAM FLATTENING

- Suppose we want to **flatten** the histogram of image **I**.
- Define the **cumulative histogram image $\mathbf{J}_1$ = P(I)**:

$$J_1(i, j) = \mathbf{P_I}[I(i, j)] = \sum_{k=0}^{I(i, j)} \mathbf{p_I}(k)$$

- At each pixel, this is the cumulative histogram evaluated at the grey level of the pixel.
- Note that: $0 \leq J_1(i, j) \leq 1$
- The elements of the **cumulative probability** image $\mathbf{J}_1$ will be **approximately** linearly distributed between 0 and 1.
- Then scale $\mathbf{J}_1$ to cover the range 0, ..., K-1, produce the histogram-flattened image **J**:
- $J(i, j) = \text{INT}\,[\,(K-1)\cdot J_1(i, j) + 0.5\,]$
- This is best understood by an example:

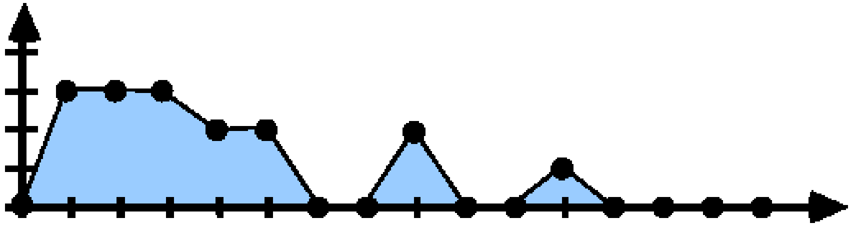
EXAMPLE

- Given a 4 x 4 image **I** with gray-level range {0, ..., 15} (K-1 = 15):

$$\mathbf{I} = \begin{array}{|c|c|c|c|} \hline 1 & 1 & 3 & 4 \\ \hline 2 & 5 & 3 & 2 \\ \hline 8 & 1 & 8 & 2 \\ \hline 4 & 5 & 3 & 11 \\ \hline \end{array}$$

- It's histogram is

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
H(k)	0	3	3	3	2	2	0	0	2	0	0	1	0	0	0	0



EXAMPLE

- The normalized histogram is

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
p(k)	0	$\frac{3}{16}$	$\frac{3}{16}$	$\frac{3}{16}$	$\frac{2}{16}$	$\frac{2}{16}$	0	0	$\frac{2}{16}$	0	0	$\frac{1}{16}$	0	0	0	0

- From which we can compute the intermediate image **J1**

$$\mathbf{I} = \begin{array}{|c|c|c|c|} \hline 1 & 1 & 3 & 4 \\ \hline 2 & 5 & 3 & 2 \\ \hline 8 & 1 & 8 & 2 \\ \hline 4 & 5 & 3 & 11 \\ \hline \end{array}$$

$$\mathbf{J}_1 = \begin{array}{|c|c|c|c|} \hline 3/16 & 3/16 & 9/16 & 11/16 \\ \hline 6/16 & 13/16 & 9/16 & 6/16 \\ \hline 15/16 & 3/16 & 15/16 & 6/16 \\ \hline 11/16 & 13/16 & 9/16 & 16/16 \\ \hline \end{array}$$

EXAMPLE

- The normalized histogram is

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
p(k)	0	$\frac{3}{16}$	$\frac{3}{16}$	$\frac{3}{16}$	$\frac{2}{16}$	$\frac{2}{16}$	0	0	$\frac{2}{16}$	0	0	$\frac{1}{16}$	0	0	0	0

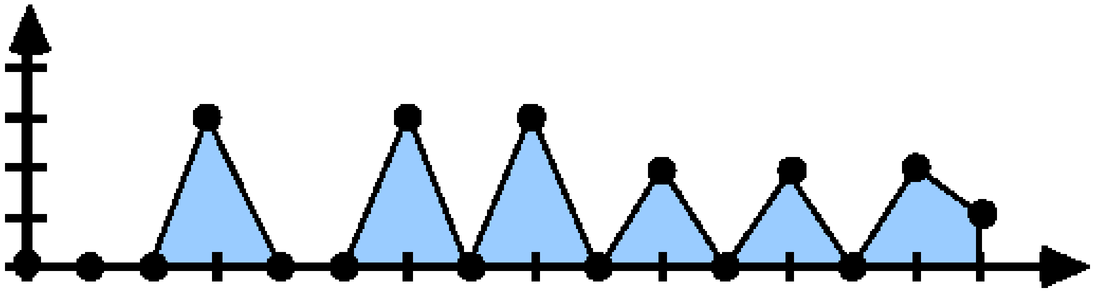
- From which we can compute the intermediate image **J1** and finally the "flattened" image **J**:

$$\mathbf{J}_1 = \begin{array}{|c|c|c|c|}\hline 3/16 & 3/16 & 9/16 & 11/16 \\ \hline 6/16 & 13/16 & 9/16 & 6/16 \\ \hline 15/16 & 3/16 & 15/16 & 6/16 \\ \hline 11/16 & 13/16 & 9/16 & 16/16 \\ \hline \end{array} \qquad \mathbf{J} = \begin{array}{|c|c|c|c|}\hline 3 & 3 & 8 & 10 \\ \hline 6 & 12 & 8 & 6 \\ \hline 14 & 3 & 14 & 6 \\ \hline 10 & 12 & 8 & 15 \\ \hline \end{array}$$

EXAMPLE

- The new, **flattened** histogram looks like this:

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
H(k)	0	0	0	3	0	0	3	0	3	0	2	0	2	0	2	1



- The heights H(k) cannot be reduced, just moved - or stacked, so:
- **Digital** histogram flattening doesn't really "flatten" the histogram - it just makes it "flatter" by **spreading out** the histogram.
- The **spaces** that appear are highly characteristic of a "flattened" histogram - especially when the original histogram is highly compressed.

Histogram Shaping

$$if\ y = F(x) \Rightarrow P_Y\ (y) = P_X(x)$$

$$P_Y(y) = P_X(x)$$

$$y = P_Y^{-1}(P_X(x))$$

HISTOGRAM SHAPING

- Can create a modified image **J** with an approximate **specified** histogram shape, such as a triangle or bell-shaped curve.
- Let $\mathbf{H_J}(k)$ be the desired histogram shape, with corresponding normalized values (probabilities) $\mathbf{p_J}(k)$.
- Define the cumulative histogram image as before

$$J_1(i, j) = \mathbf{P_I}[I(i, j)] = \sum_{k=0}^{I(i, j)} \mathbf{p_I}(k)$$

- We also define the cumulative probabilities:

$$\mathbf{P_J}(n) = \sum_{k=0}^{n} \mathbf{p_J}(k)$$

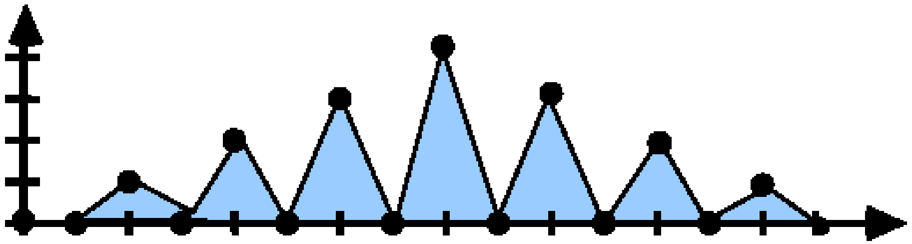
EXAMPLE

- Consider the same image as in the last example. We had

$$\mathbf{I} = \begin{bmatrix} 1 & 1 & 3 & 4 \\ 2 & 5 & 3 & 2 \\ 8 & 1 & 8 & 2 \\ 4 & 5 & 3 & 11 \end{bmatrix} \qquad \mathbf{J}_1 = \begin{bmatrix} 3/16 & 3/16 & 9/16 & 11/16 \\ 6/16 & 13/16 & 9/16 & 6/16 \\ 15/16 & 3/16 & 15/16 & 6/16 \\ 11/16 & 13/16 & 9/16 & 16/16 \end{bmatrix}$$

- Fit this t

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$H_{\mathbf{J}}(k)$	0	0	1	0	2	0	3	0	4	0	3	0	2	0	1	0
$p_{\mathbf{J}}(k)$	0	0	$\frac{1}{16}$	0	$\frac{2}{16}$	0	$\frac{3}{16}$	0	$\frac{4}{16}$	0	$\frac{3}{16}$	0	$\frac{2}{16}$	0	$\frac{1}{16}$	0



EXAMPLE

- Here's the cumulative (summed) probabilities associated with it:

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$\mathbf{P_J}(n)$	0	0	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{3}{16}$	$\frac{3}{16}$	$\frac{6}{16}$	$\frac{6}{16}$	$\frac{10}{16}$	$\frac{10}{16}$	$\frac{13}{16}$	$\frac{13}{16}$	$\frac{15}{16}$	$\frac{15}{16}$	$\frac{16}{16}$	$\frac{16}{16}$

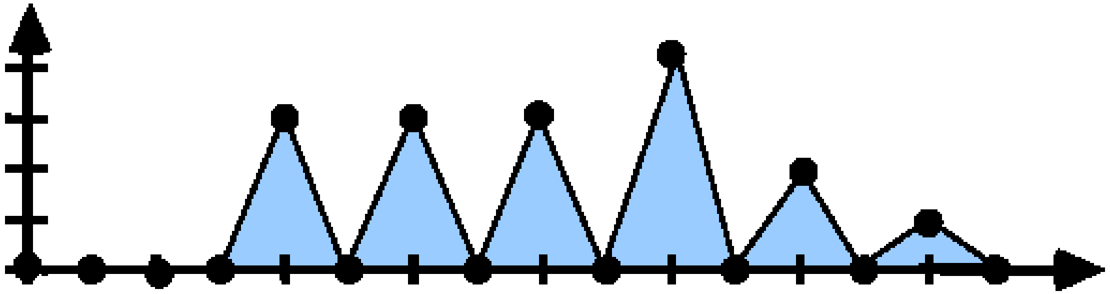
- **Careful** visual inspection of $\mathbf{J}_1$ let's us form the new image:

$$\mathbf{J} = \begin{array}{|c|c|c|c|}\hline 4 & 4 & 8 & 10 \\ \hline 6 & 10 & 8 & 6 \\ \hline 12 & 4 & 12 & 6 \\ \hline 10 & 10 & 8 & 14 \\ \hline\end{array}$$

EXAMPLE

- Here's the new histogram:

k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$H_J(k)$	0	0	0	0	3	0	3	0	3	0	4	0	2	0	1	0



HISTOGRAM MATCHING

- Just a special case of histogram shaping.
- Difference: the histogram of the original image **I** is matched to that of another image **I´** .
- Otherwise the procedure is identical, once the cumulative probabilities are computed for the model image **I´** .
- Useful application: **Comparing** similar images of the same scene obtained under different conditions (e.g., lighting, time of day). Extends the concept of equalizing AOD described earlier.

BASIC ALGEBRAIC IMAGE OPERATIONS

- Algebraic image operations (between images) are quite simple
- Suppose we have two N x N images $\mathbf{I_1}$ and $\mathbf{I_2}$. The four basic algebraic operations (like the ones on your calculator) are:
- **Pointwise Matrix Addition**
- $\mathbf{J} = \mathbf{I_1} + \mathbf{I_2}$ if $J(i, j) = \mathbf{I_1}(i, j) + \mathbf{I_2}(i, j)$ for $0 <= i, j <= N-1$
- **Pointwise Matrix Subtraction**
- $\mathbf{J} = \mathbf{I_1} - \mathbf{I_2}$ if $J(i, j) = \mathbf{I_1}(i, j) - \mathbf{I_2}(i, j)$ for $0 <= i, j <= N-1$
- **Pointwise Matrix Multiplication**
- $\mathbf{J} = \mathbf{I_1} .* \mathbf{I_2}$ if $J(i, j) = \mathbf{I_1}(i, j) \times \mathbf{I_2}(i, j)$ for $0 <= i, j <= N-1$
- **Pointwise Matrix Division**
- $\mathbf{J} = \mathbf{I_1} ./ \mathbf{I_2}$ if $J(i, j) = \mathbf{I_1}(i, j) / \mathbf{I_2}(i, j)$ for $0 <= i, j <= N-1$

APPLICATIONS OF ALGEBRAIC OPERATIONS

- Although simple, algebraic operations form the backbone of most of digital image processing
- We will look at two simple but **important** applications of algebraic operations on images:
  - Frame averaging for noise reduction
  - Motion detection

Frame Averaging for Noise Reduction

- An image **J** is often corrupted by **additive noise**:
  - Surface radiation scatter
  - Noise in the camera
  - Thermal noise in a computer circuit
  - Channel transmission noise
- We can model such a **noisy image** as the sum of **original, uncorrupted image I** and a **noise image N**:

$$\mathbf{J} = \mathbf{I} + \mathbf{N},$$

- where the elements N(i, j) of **N** are **random variables**

Frame Averaging

- We will assume that the noise is **zero mean** (ergodic), which means that the **sample mean** of M noise matrices tends towards zero as M grows large:

$$\left(\frac{1}{M}\right)\sum_{i=1}^{M} \mathbf{N}_i = \left(\frac{1}{M}\right)\left[\, \mathbf{N}_1 + \cdots + \mathbf{N}_M \,\right] \simeq \mathbf{0} \ \ \textbf{(matrix of zeros)}$$

- Averaging together large many zero-mean noise samples produces a value near zero

Frame Averaging

- Suppose that we obtain M images J_1, ..., $\mathbf{J}_M$ of the **same scene**
- - In rapid succession, so that there is **no motion** between frames
- - Or there is **no motion** in the scene.
- However, the frames are **noisy:**

$$\mathbf{J}_i = \mathbf{I}_i + \mathbf{N}_i \text{ for } i = 1, \ldots, M.$$

- Suppose that we **average** the frames together:

$$\mathbf{J} = \left(\frac{1}{M}\right)\sum_{i=1}^{M} \mathbf{J}_i = \left(\frac{1}{M}\right)\sum_{i=1}^{M}\left[\mathbf{I}_i + \mathbf{N}_i\right] = \left(\frac{1}{M}\right)\sum_{i=1}^{M}\mathbf{I}_i + \left(\frac{1}{M}\right)\sum_{i=1}^{M}\mathbf{N}_i$$

Frame Averaging

- However, since $\mathbf{I}_1 = \mathbf{I}_2 = \cdots = \mathbf{I}_M = \mathbf{I}$, then

$$\left(\frac{1}{M}\right)\sum_{i=1}^{M} \mathbf{I}_i = \left(\frac{1}{M}\right)[\,\mathbf{I} + \mathbf{I} + \cdots + \mathbf{I}\,] = \left(\frac{1}{M}\right)\cdot M \cdot \mathbf{I} = \mathbf{I}$$

- and from before

$$\left(\frac{1}{M}\right)\sum_{i=1}^{M} \mathbf{N}_i \sim= \mathbf{0}$$

- Hence we can expect that

$$\mathbf{J} \sim= \mathbf{I} + \mathbf{0} \sim= \mathbf{I}$$

- if enough frames (M) are averaged together

Motion Detection

- Often it is of interest to detect **object motion** between frames
- **Applications**: video compression, target recognition and tracking, security cameras, surveillance, automated inspection, etc.
- Here is a **simple** approach:
- Let $\mathbf{I}_1$, $\mathbf{I}_2$ be consecutive frames taken in close time proximity, e.g., from a video camera
- Form the absolute difference image

$$\mathbf{J} = |\mathbf{I}_1 - \mathbf{I}_2|$$

- Applying a **full-scale contrast stretch** to **J** will give a more visually dramatic result

Basic Geometric Image Operations

- Geometric image operations are the **opposite** of point operations: they modify **spatial positions** of pixels but not **gray levels**
- A geometric operation generally requires **two steps:**
- (1) A **spatial mapping of image coordinates** giving a new image function **J**: J(i, j) = I(i´ , j´) = I[a(i, j), b(i, j)]
- The coordinates a(i, j) and b(i, j) **are not generally integers !**
- For example: a(i, j) = i/3.5, b(i, j) = j/4.5
- Then J(i, j) = I(i/3.5, j/4.5), which has undefined coordinates!
- Which element of **I** do we define J(i, j) to be?

INTERPOLATION

- Thus implies the need for a second operation:
- (2) **Interpolate** non-integer coordinates a(i, j) and b(i, j) to integer values, so that **J** can be expressed in **row-column format**

Nearest Neighbor Interpolation

- Simple-minded
- The geometrically transformed coordinates are mapped to the **nearest integer coordinates**:

  J(i, j) = I **{** INT[a(i, j)+0.5], INT[b(i, j)+0.5]**}**
- Serious drawback: Sudden intensity changes lead to the "jagged edge" effect

The Basic Geometric Transformations

- The most basic geometric transformations are
- - Translation
- - Rotation
- - Zooming

Translation

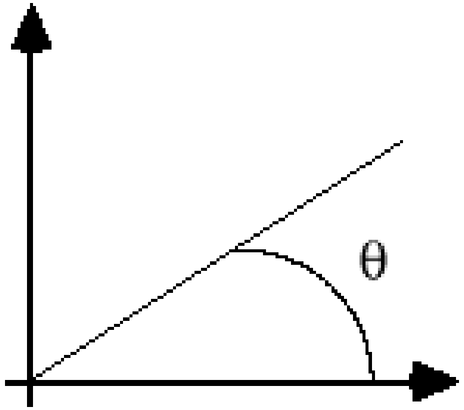
- Translation is the simplest geometric operation and requires no interpolation
- Let $a(i, j) = i - i_0$, $b(i, j) = j - j_0$ where (i_0 , j_0) are **constants**
- In this case $J(i, j) = I(i - i_0 , j - j_0)$; a **shift** or translation of the image by an amount i_0 in the vertical (row) direction and an amount j_0 in the horizontal direction

Rotation

- Rotation of an image by an angle **q** relative to the x-axis is accomplished by the following transformation:

$$a(i, j) = i \cos(\theta) - j \sin(\theta)$$

$$b(i, j) = i \sin(\theta) + j \cos(\theta)$$



- Simplest cases:

$$\theta = 90° : [a(i, j), b(i, j)] = (-j, i)$$

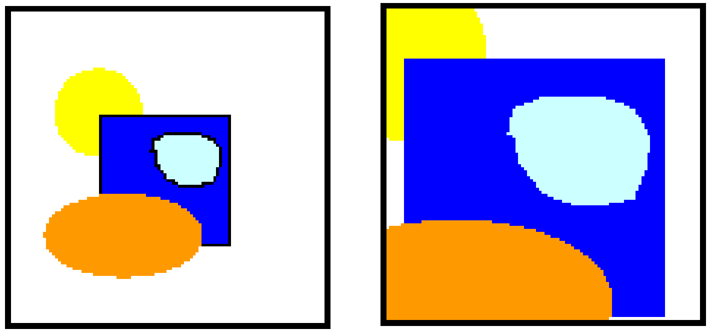
$$\theta = 180° : [a(i, j), b(i, j)] = (-i, -j)$$

$$\theta = -90° : [a(i, j), b(i, j)] = (j, -i)$$

Zooming

- Zooming **magnifies** an image by the mapping functions

  a(i, j) = i / c and b(i, j) = j / d
- where c >= 1 and d >= 1



original 2x zoomed

- For large magnifications, the zoomed image will look "blotchy" if a simple nearest neighbor interpolation is used