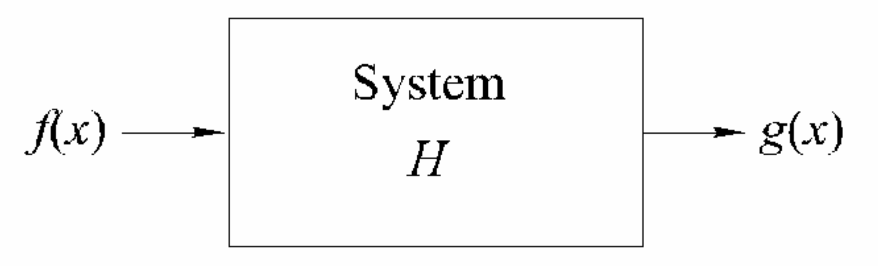
# Image Enhancement

Part 1: pixel-based operations

# Review: Linear Systems

- We define a system as a unit that converts an input function into an output function

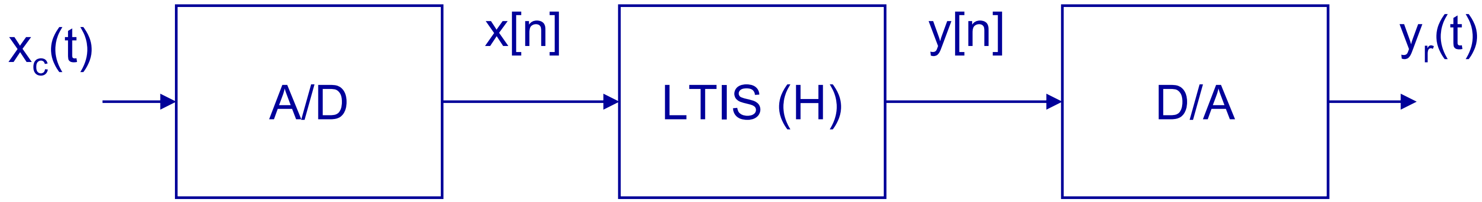



$$g(x) = H[f(x)]$$

Independent variable

System operator or Transfer function

# Linear Time Invariant Discrete Time Systems




$$Y(e^{j\omega}) = H(e^{j\omega})X(e^{j\omega})$$

$$Y(\omega) = H(\omega)X(\omega) \leftrightarrow y[n] = h[n] * x[n]$$

$$H(j\Omega) = \begin{cases} H(j\Omega) & |\Omega| < \pi / T \\ 0 & |\Omega| \geq \pi / T \end{cases}$$

IF
- The input signal is bandlimited
- The Nyquist condition for sampling is met
- The digital system is linear and time invariant

THEN
The overall continuous time system is equivalent to a LTIS whose frequency response is H.

# Overview of Linear Systems

- Let $g_i(x) = H[f_i(x)]$

  where $f_i(x)$ is an arbitrary input in the class of all inputs $\{f(x)\}$, and $g_i(x)$ is the corresponding output.

- If

$$H\{a \cdot f[n] + b \cdot g[n]\} = aH\{f[n]\} + bH\{g[n]\}$$

  Then the system $H$ is called a *linear system*.

- A linear system has the properties of *additivity* and *homogeneity*.

# Linear Systems

- The system H is called *shift invariant* if

$$g_i(x) = H[f_i(x)] \text{ implies that } g_i(x + x_0) = H[f_i(x + x_0)]$$

for all $f_i(x) \in \{f(x)\}$ and for all $x_0$.

- This means that offsetting the independent variable of the input by $x_0$ causes the same offset in the independent variable of the output. Hence, the input-output relationship remains the same.

# Linear Systems

- The operator *H* is said to be *causal*, and hence the system described by *H* is a *causal system*, if there is no output before there is an input.
- In other words

$$f(x) = 0 \text{ for } x < x_0 \text{ implies that } g(x) = H[f(x)] = 0 \text{ for } x < x_0.$$

- A linear system *H* is said to be *stable* if its response to any *bounded* input is *bounded*. That is, if

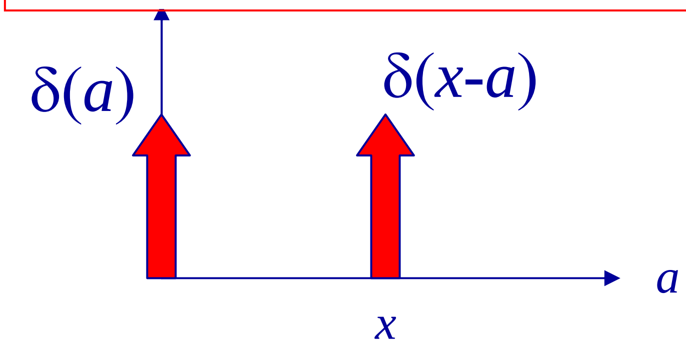
$$|f(x)| < K \text{ implies that } |g(x)| < cK$$

where *K* and *c* are constants.

# Linear Systems

- A *unit impulse function*, denoted δ(*a*), is *defined* by the expression

$$\int_{-\infty}^{\infty} f(a)\delta(x-a)da = f(x).$$




- The response of a system to a unit impulse function is called the *impulse response* of the system.

$$h(x) = H[\delta(x)]$$

$$h[n] = H\{\delta[n]\}$$

# Linear Systems

- If *H* is a linear shift-invariant system, then we can find its response to any input signal f(x) as follows:

$$g(x) = \int_{-\infty}^{\infty} f(\alpha)h(x-\alpha)d\alpha.$$

$$g[n] = \sum_{k=-\infty}^{+\infty} f[k]h[n-k]$$

- Underlying model: signal="sum" of deltas of amplitude f[n]
- This expression is called the *convolution integral*. It states that the response of a linear, fixed-parameter system is completely characterized by the convolution of the input with the system impulse response.

# Linear Systems

- Convolution of two functions of a continuous variable is defined as

$$f(x) * h(x) = \int_{-\infty}^{\infty} f(\alpha) h(x - \alpha) d\alpha$$

- In the discrete case

$$f[n] * h[n] = \sum_{m=-\infty}^{\infty} f[m] h[n - m]$$

# Linear Systems

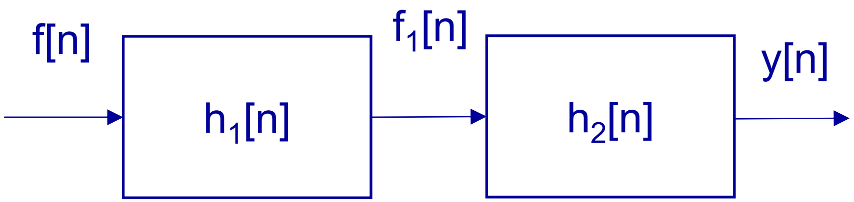
- In the 2D discrete case

$$f[n_1, n_2] * h[n_1, n_2] = \sum_{m_1=-\infty}^{\infty} \sum_{m_2=-\infty}^{\infty} f[m_1, m_2] h[n_1 - m_1, n_2 - m_2]$$

$h[n_1, n_2]$ is a linear filter.

# Linear systems

- Cascade ("in serie")




$$h[n] = h_1[n] * h_2[n]$$

$$H(\Omega) = H_1(\Omega) H_2(\Omega)$$

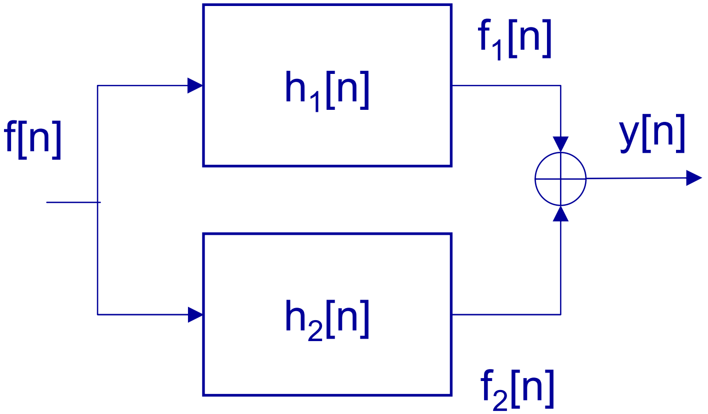
$$H[k] = H_1[k] H_2[k]$$

Proof

$$y[n] = f_1[n] * h_2[n] = f[n] * h_1[n] * h_2[n] =$$

$$= f[n] * (h_1[n] * h_2[n])$$

- Parallel ("in parallelo")




$$h[n] = h_1[n] + h_2[n]$$

$$H(\Omega) = H_1(\Omega) + H_2(\Omega)$$

$$H[k] = H_1[k] + H_2[k]$$

# IP Algorithms

## Spatial domain

- Operations are performed in the image domain
- Image ⇔ matrix of numbers
- Examples
  - luminance adaptation
  - chromatic adaptation
  - contrast enhancement
  - spatial filtering
  - edge detection
  - noise reduction

## Transform domain

- Some operators are used to project the image in another space
- Operations are performed in the transformed domain
  - Fourier (DCT, FFT)
  - Wavelet (DWT,CWT)
- Examples
  - coding
  - denoising
  - image analysis

Most of the tasks can be implemented both in the image and in the transformed domain. The choice depends on the context and the specific application.

# Spatial domain processing

## Pixel-wise

- Operations involve the single pixel
- Operations:
  - histogram equalization
  - change of the colorspace
  - addition/subtraction of images
  - get negative of an image
- Applications:
  - luminance adaptation
  - contrast enhancement
  - *chromatic adaptation*

## Local-wise

- The neighbourhood of the considered pixel is involved
  - Any operation involving digital filters is local-wise
- Operations:
  - correlation
  - convolution
  - filtering
  - transformation
- Applications
  - smoothing
  - sharpening
  - noise reduction
  - edge detection

# Image enhancement

- There is no general unifying theory of image enhancement at present because there is no general standard of image quality that can serve as a design criterion for an image enhancement processor.
  - Consideration is given here to a variety of techniques that have proved useful for human observation improvement and image analysis.
- [Pratt, Chapter 10]

# Pixel-wise operations

- Contrast enhancement
  - Amplitude scaling
  - Histogram straching/shrinking, sliding, equalization
- Contrast can often be improved by amplitude rescaling of each pixel

# Amplitude scaling

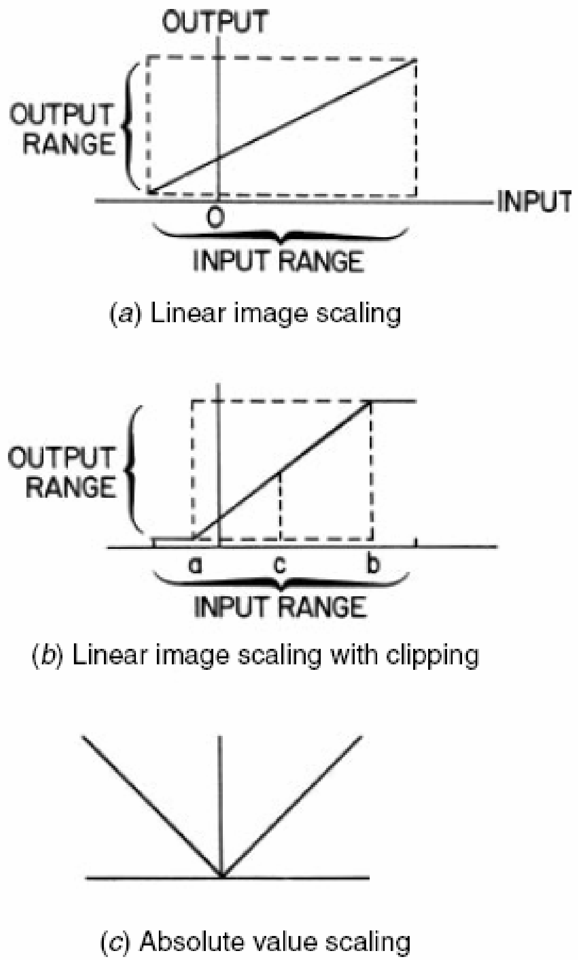



**FIGURE 10.1-2.** Image scaling methods.

*Window-level transformation*. The window value is the width of the linear slope; the level is located at the midpoint *c* of the slope line. Very common in medical imaging.

# Amplitude scaling

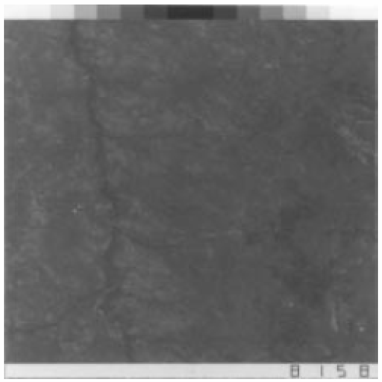
Q component of a YIQ image representation.


(*a*) Linear, full range, – 0.147 to 0.169


(*b*) Clipping, 0.000 to 0.169

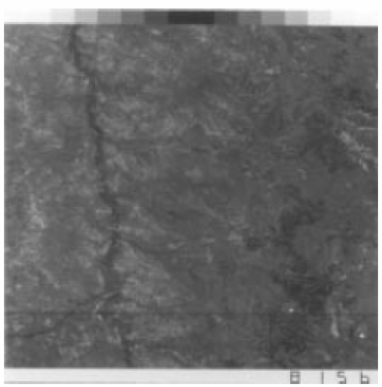

(*c*) Absolute value, 0.000 to 0.169

# Window level transformation: ex.

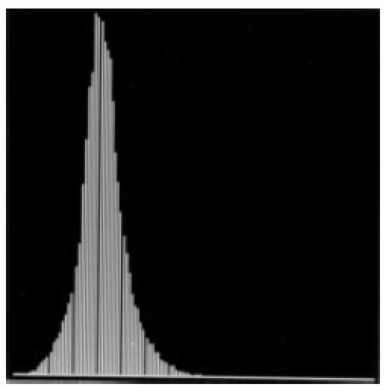


(*a*) Original

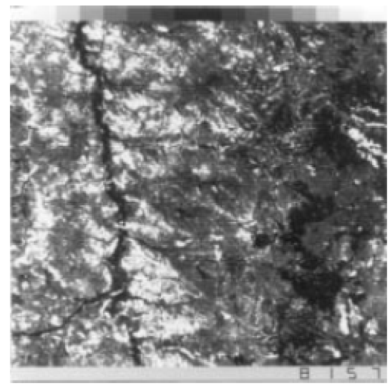
(*b*) Original histogram



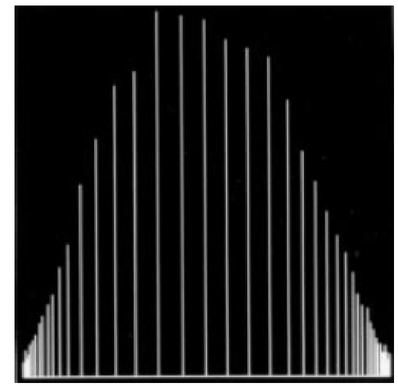
(*c*) Min. clip = 0.17, max. clip = 0.64



(*d*) Enhancement histogram



(*e*) Min. clip = 0.24, max. clip = 0.35



(*f*) Enhancement histogram

**FIGURE 10.1-4.** Window-level contrast stretching of an earth satellite image.

Gray scale contouring is at the threshold of visibility.

# Contrast enhancement via graylevel transf.

- Point transformations that modify the contrast of an image within a display's dynamic range
- Often nonlinear point transformations

$$G[j,k] = \left(F[j,k]\right)^p$$

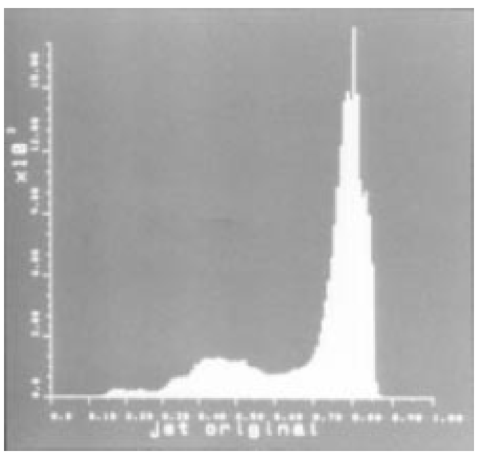
$$0 \le F[j,k] \le 1$$

$$p: \text{ power law varaible}$$

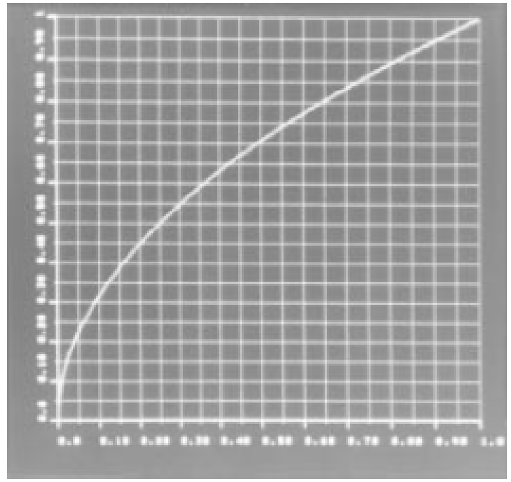
# example

original


(*a*) Original

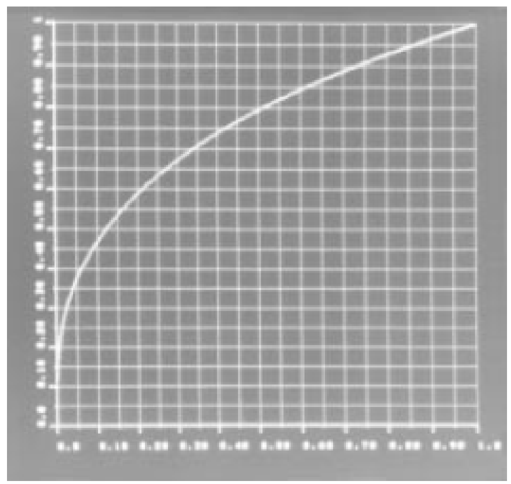



(*b*) Original histogram



(*a*) Square root function


(*b*) Square root output



(*c*) Cube root function

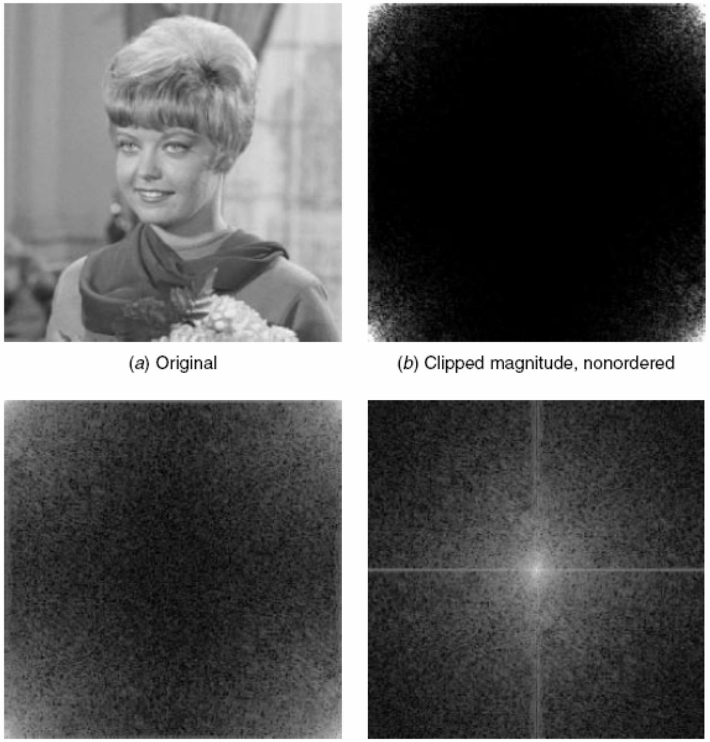

(*d*) Cube root output

# log amplitude scaling

- The logarithm function is useful for scaling image arrays with a very wide dynamic range.

$$G(j,k) = \frac{\log_e\{1.0 + aF(j,k)\}}{\log_e\{2.0\}}$$

$a>0$



(*a*) Original

(*b*) Clipped magnitude, nonordered

(*c*) Log magnitude, nonordered

(*d*) Log magnitude, ordered

# Reverse and Inverse functions

- Reverse function

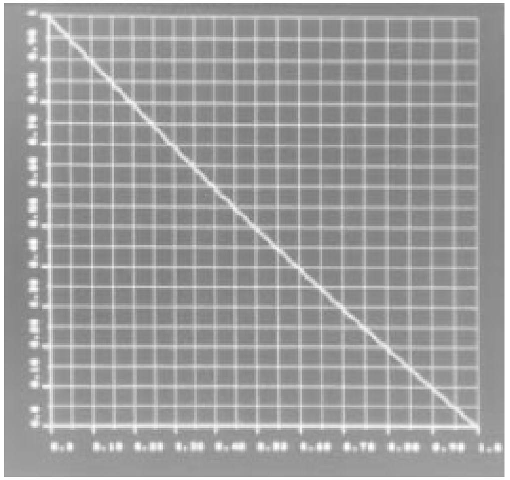
$$G[i,k] = (1 - F[i,k])$$
$$0 \le F[i,k] \le 1$$

- Inverse function

$$G(j,k) = \begin{cases} 1.0 & \text{for } 0.0 \le F(j,k) < 0.1 \\ \dfrac{0.1}{F(j,k)} & \text{for } 0.1 \le F(j,k) \le 1.0 \end{cases}$$

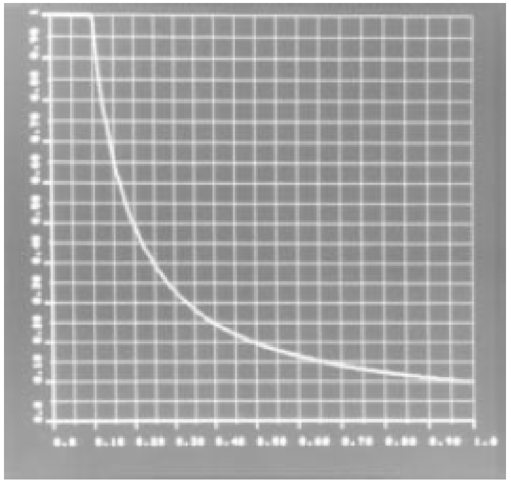
clipped below 0.1 to maintain the range (max value=1)

# example



(*a*) Reverse function


(*b*) Reverse function output



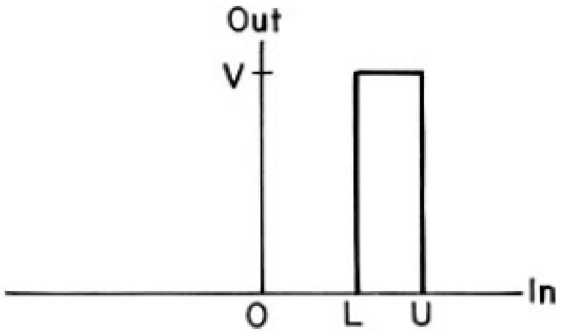
(*c*) Inverse function


(*d*) Inverse function output

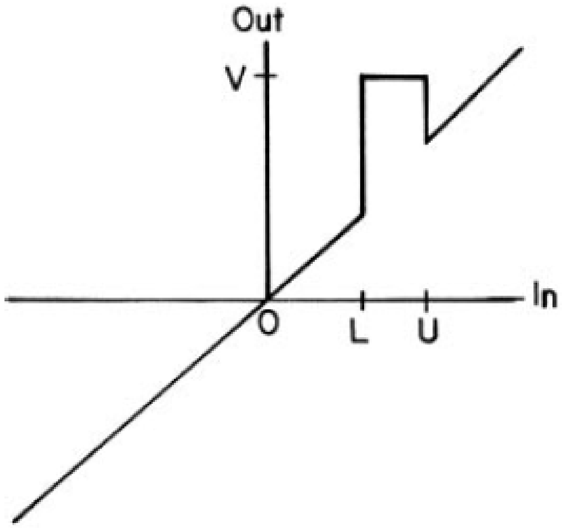
# Level slicing

- Pixels within the amplitude passband are rendered maximum white in the output, and pixels outside the passband are rendered black.

- Pixels outside the amplitude passband are displayed in their original state



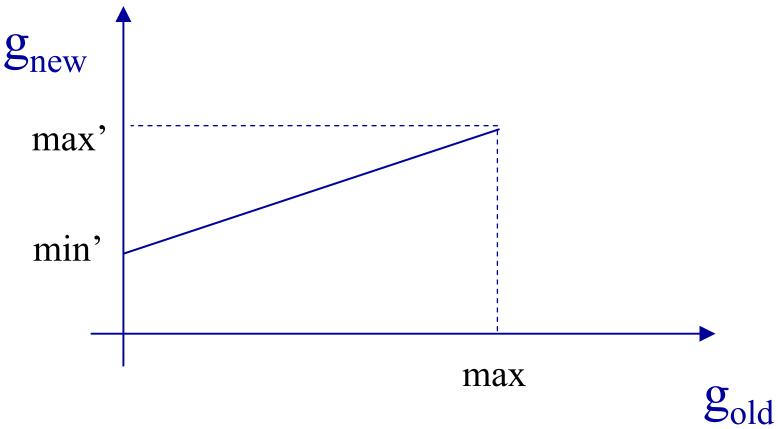
(a) Zero background scaling transformation

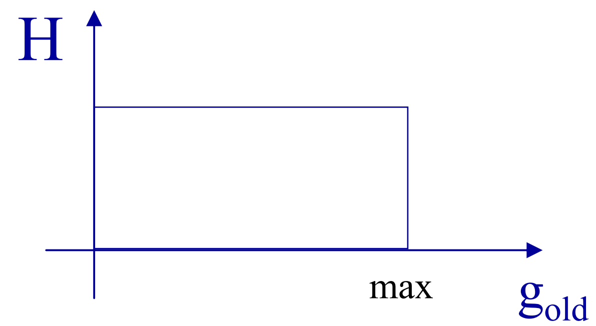


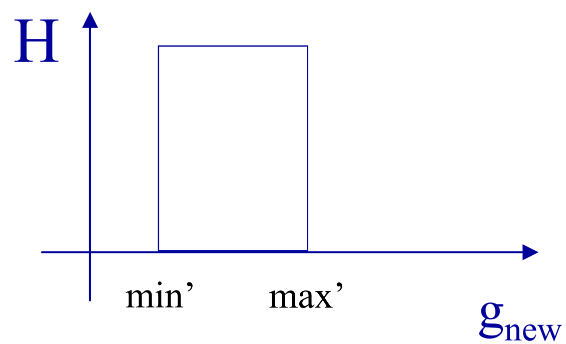
(b) Image background scaling transformation

# Histogram changes

- Graylevel transformations induce histogram changes



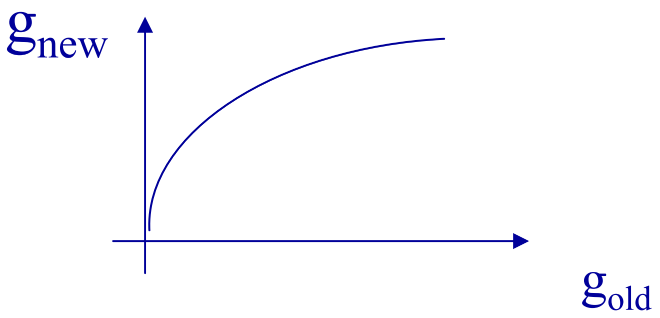



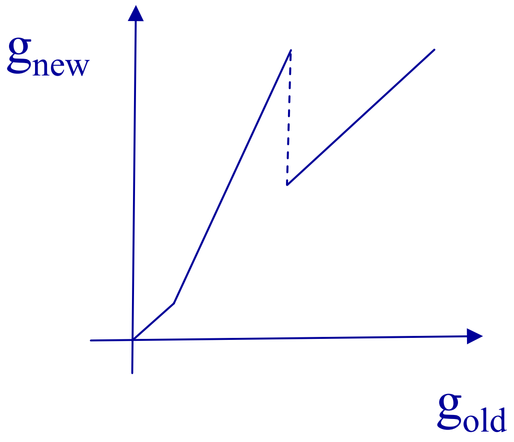



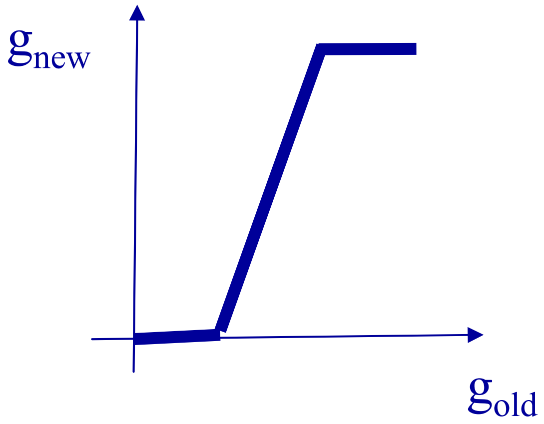
# Other non-linear transformations

- Used to emphasize mid-range levels

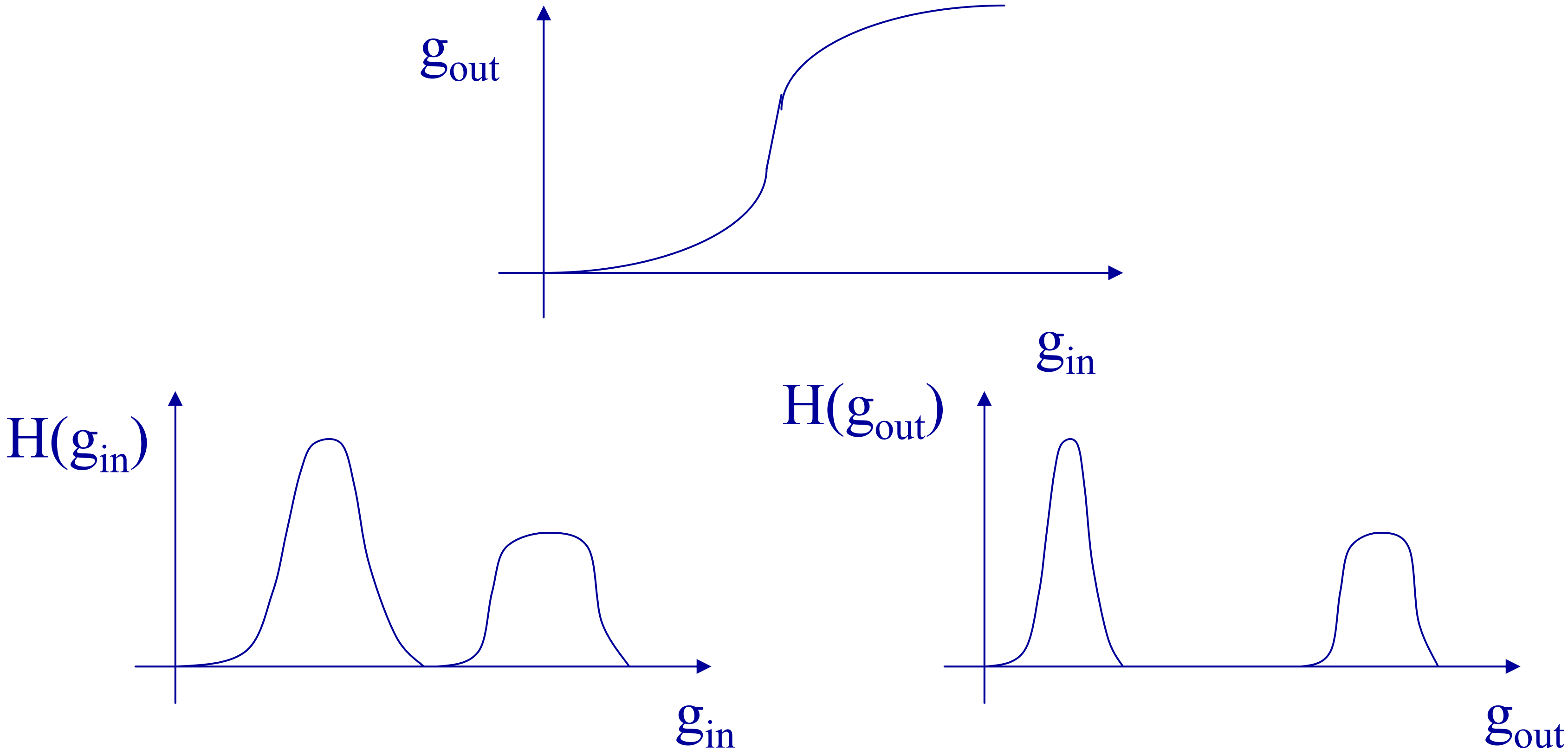
$g_{new} = g_{old} + g_{old}\,C\,(g_{old,max} - g_{old})$



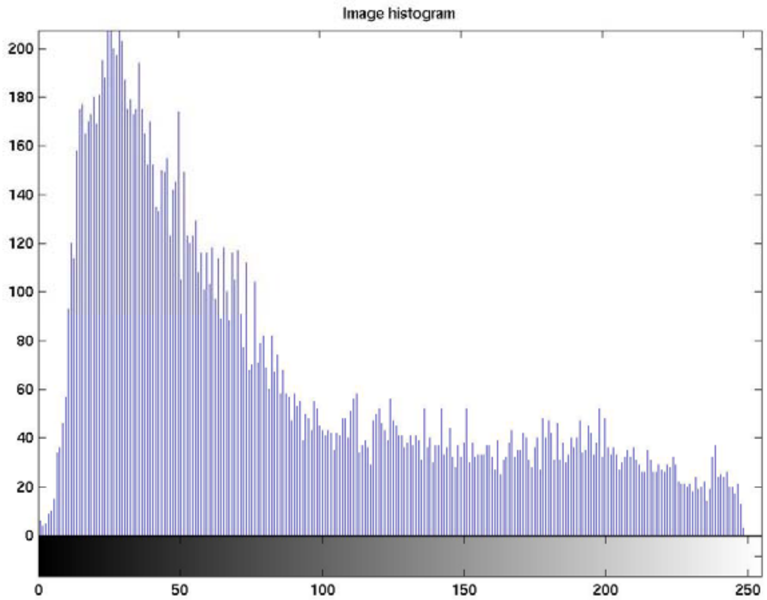



# Sigmoid transformation (*soft thresholding*)



# Pixel-wise: Histogram equalization

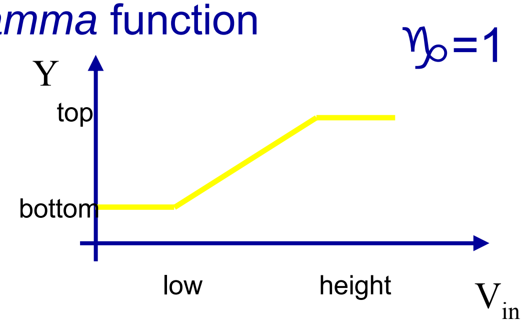
- Pixel features: luminance, color,
- Histogram equalization: shapes the intensity histogram to approximate a specified distribution
  - It is often used for enhancing contrast by shaping the image histogram to a uniform distribution over a given number of grey levels. The grey values are redistributed over the dynamic range to have a constant number of samples in each interval (i.e. histogram bin).
  - Can also be applied to colormaps of color images.

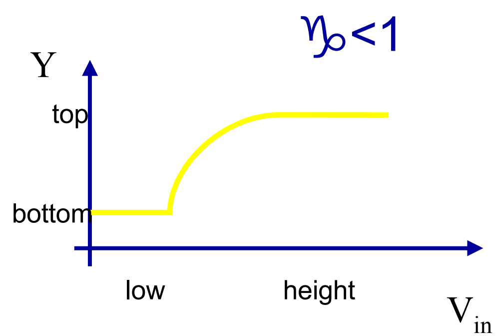


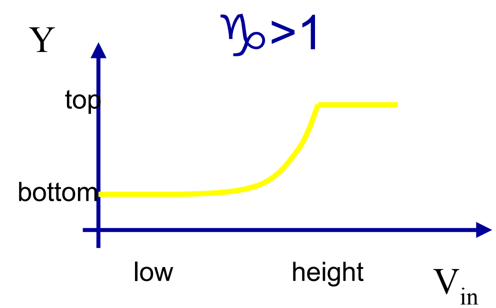
# Histogram equalization

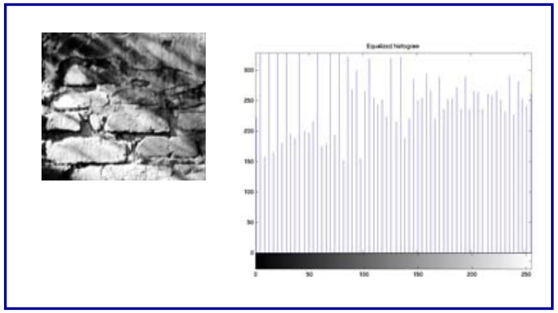
Can be used to compensate the distortions in the gray level distribution due to the non-linearity of a system component

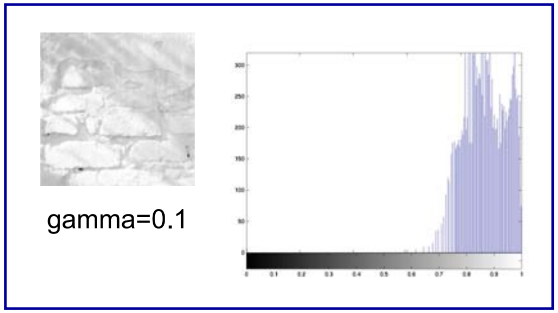
*Gamma* function

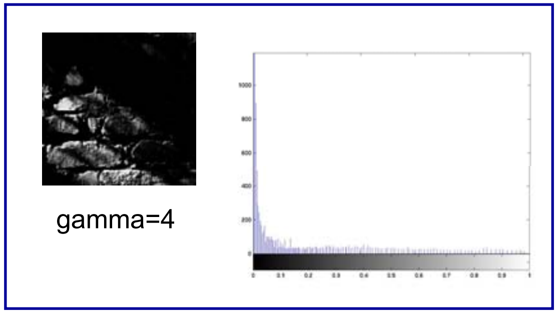


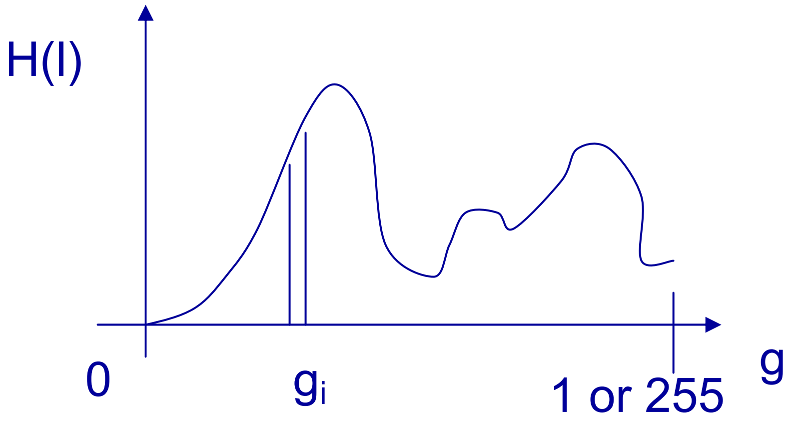





# Histogram

- Function H=H(g) indicating the number of pixels having gray-value equal to g
  - Non-normalized images: $0 \le g \le 255 \rightarrow$ bin-size$\ge$1, can be integer
  - Normalized images: $0 \le g \le 1 \rightarrow$ bin-size<1




$$A = \int_0^{\max} H(g)dg$$

area under the curve=number of pixels

$$A = \sum_{i=1}^{N_g} H[g_i]$$

In the continuous case

$$H(g) = -\frac{dA(g)}{dg} = \lim_{\Delta g \to 0} \frac{A(g) - A(g+\Delta g)}{\Delta g}$$

# Histogram transformation

$g_{out} = f(g_{in}) \Rightarrow g_{in} = f^{-1}(g_{out}), \quad f$ non-decreasing function

$H(g_{in}) \Rightarrow H(g_{out}), \quad$ namely

$$H(g_{out}) = \frac{H[f^{-1}(g_{out})]}{f'[f^{-1}(g_{out})]}, \quad f' = \frac{\partial f}{\partial g}$$

# More formally

- The histogram modification process can be considered to be a monotonic point transformation $g_d=T\{f_c\}$ for which the input amplitude variable $f_1 \leq f_c \leq f_C$ is mapped into an output variable $g_1 \leq g_d \leq g_D$ such that the output probability distribution $Pr\{g_d=b_d\}$ follows some desired form for a given input probability distribution $Pr\{f_c=a_c\}$ where $a_c$ and $b_d$ are reconstruction values of the $c^{th}$ and $d^{th}$ levels.
  - Clearly, the input and output probability distributions must each sum to unity.

$$\sum_{c=1}^{C} P_R\{f_c = a_c\} = 1$$

$$\sum_{d=1}^{D} P_R\{g_d = b_d\} = 1$$

NB: C and D are caps!

# Histogram equalization

– Furthermore, the cumulative distributions must equate for any input index *c*.
  - the probability that pixels in the input image have an amplitude less than or equal to $a_c$ must be equal to the probability that pixels in the output image have amplitude less than or equal to $b_d$, where $b_d = T\{a_c\}$ because the transformation is monotonic. Hence

$$\sum_{n=1}^{d} P_R\{g_n = b_n\} = \sum_{m=1}^{c} P_R\{f_m = a_m\} \qquad \text{(a)}$$

cumulative probability distribution of the output

cumulative probability distribution of the input

$$P_f(f) \approx \sum_{m=1}^{c} H_F(m)$$

histogram

  - in the continuous domain (easier for calculations)

$$\int_{g_{\min}}^{g} p_g(g)\,dg = \int_{f_{\min}}^{f} pf(f)\,df$$

$p_f(f)$ and $p_g(g)$ are the probability densities of f and g

NB: C and D are lowercase!

# Histogram equalization

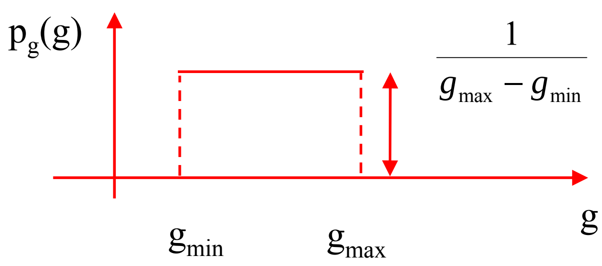
cumulative probability distribution of the output

(a) $$\sum_{n=1}^{d} P_R\{g_n = b_n\} = \sum_{m=1}^{c} H_F(m)$$ $$\int_{g_{min}}^{g} p_g(g)\,dg = P_f(f)$$ (b)

cumulative histogram

When the output density is forced to be the uniform density

$$p_g(g) = \frac{1}{g_{max} - g_{min}} \quad \text{(area=1)}$$

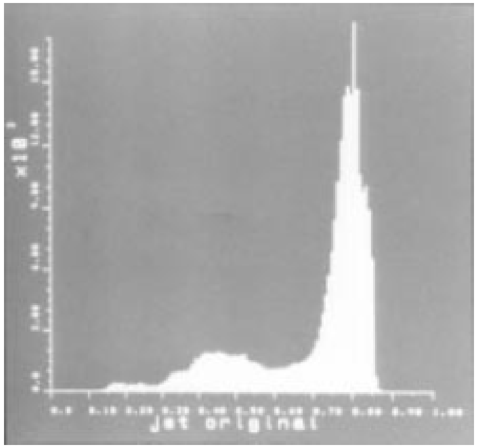



Solving (b) for g we get the histogram equalization transfer function:

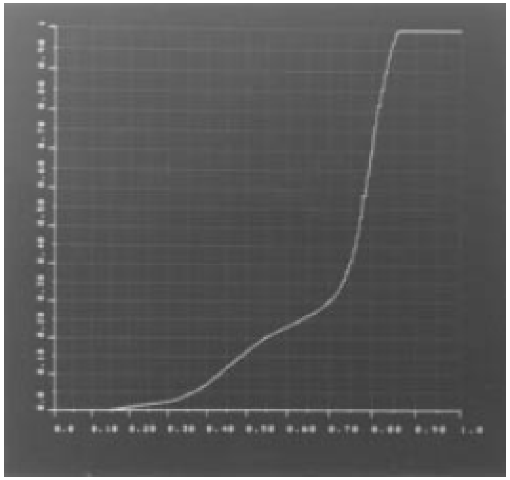
$$g = (g_{max} - g_{min})P_f(f) + g_{min}$$

# example


(*a*) Original




(*b*) Original histogram



(*b*) Transfer function


(*c*) Histogram equalized

# Some mappings

**TABLE 10.2-1. Histogram Modification Transfer Functions**

| Output Probability Density Model | | Transfer Function[a] |
|---|---|---|
| Uniform | $p_g(g) = \frac{1}{g_{max} - g_{min}} \quad g_{min} \le g \le g_{max}$ | $g = (g_{max} - g_{min}) P_f(f) + g_{min}$ |
| Exponential | $p_g(g) = \alpha \exp\{-\alpha(g - g_{min})\} \quad g \le g_{min}$ | $g = g_{min} - \frac{1}{\alpha} \ln\{1 - P_f(f)\}$ |
| Rayleigh | $p_g(g) = \frac{g - g_{min}}{\alpha^2} \exp\left\{-\frac{(g - g_{min})^2}{2\alpha^2}\right\} \quad g \ge g_{min}$ | $g = g_{min} + \left[2\alpha^2 \ln\left\{\frac{1}{1 - P_f(f)}\right\}\right]^{1/2}$ |
| Hyperbolic (Cube root) | $p_g(g) = \frac{1}{3} \frac{g^{-2/3}}{g_{max}^{1/3} - g_{min}^{1/3}}$ | $g = \left[g_{max}^{1/3} - g_{min}^{1/3}[P_f(f)] + g_{max}^{1/3}\right]^3$ |
| Hyperbolic (Logarithmic) | $p_g(g) = \frac{1}{g[\ln\{g_{max}\} - \ln\{g_{min}\}]}$ | $g = g_{min}\left(\frac{g_{max}}{g_{min}}\right)^{P_f(f)}$ |

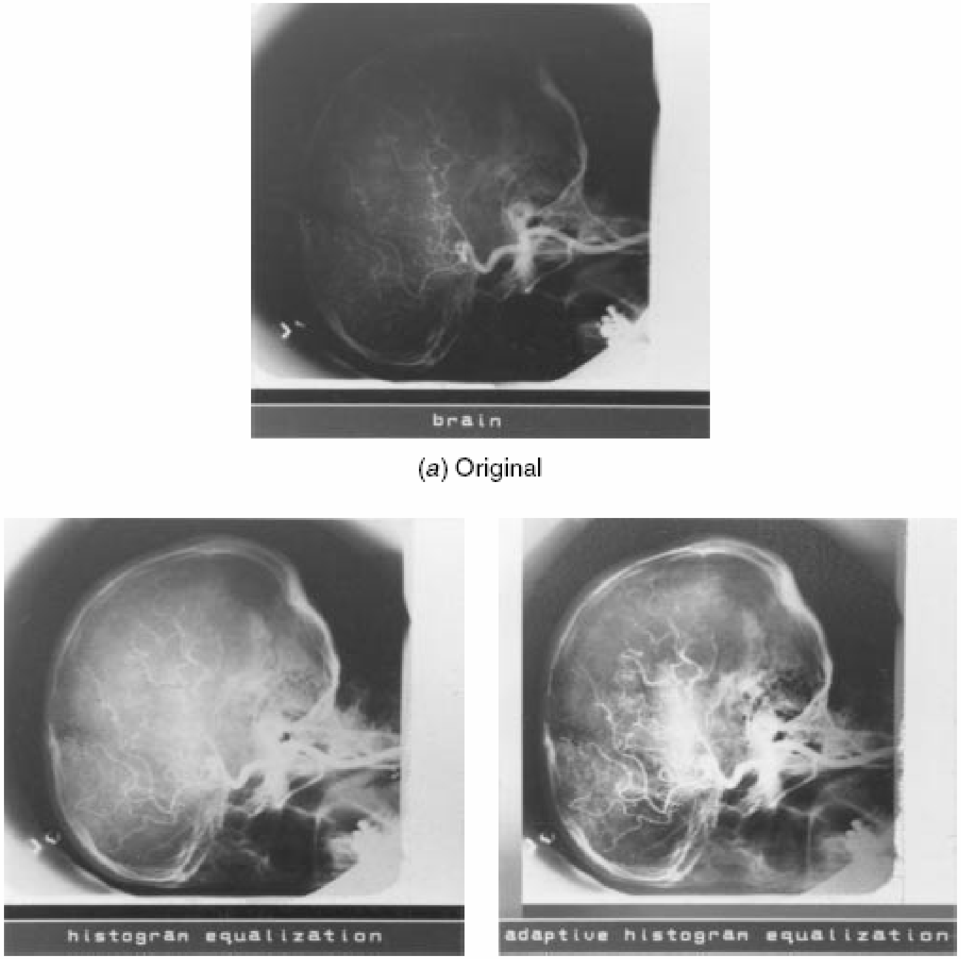
[a]The cumulative probability distribution $P_f(f)$, of the input image is approximated by its cumulative histogram:

$$p_f(f) \approx \sum_{m=0}^{j} H_F(m)$$

# Adaptive hist. equalization

- The mapping function can be made *spatially adaptive* by applying histogram modification to each pixel based on the histogram of pixels *within a moving window neighborhood*.
  - This technique is obviously computationally intensive, as it requires histogram generation, mapping function computation, and mapping function application at each pixel.
  - Some interpolation-based solutions can be envisioned to improve computational efficiency

# example

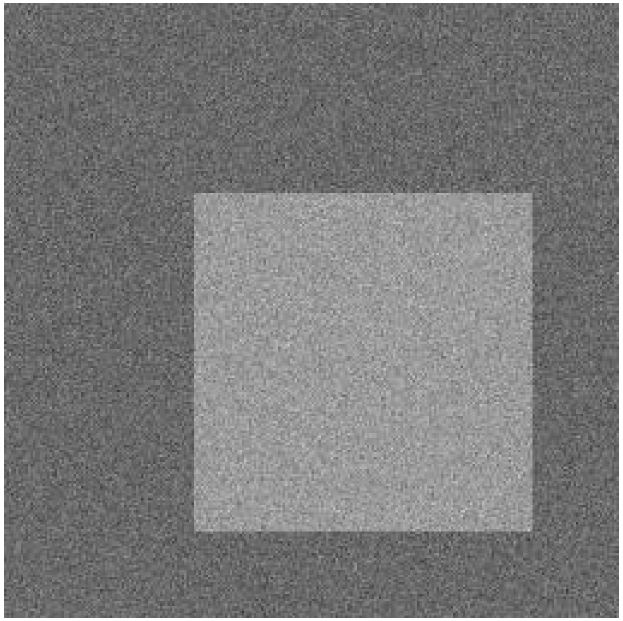


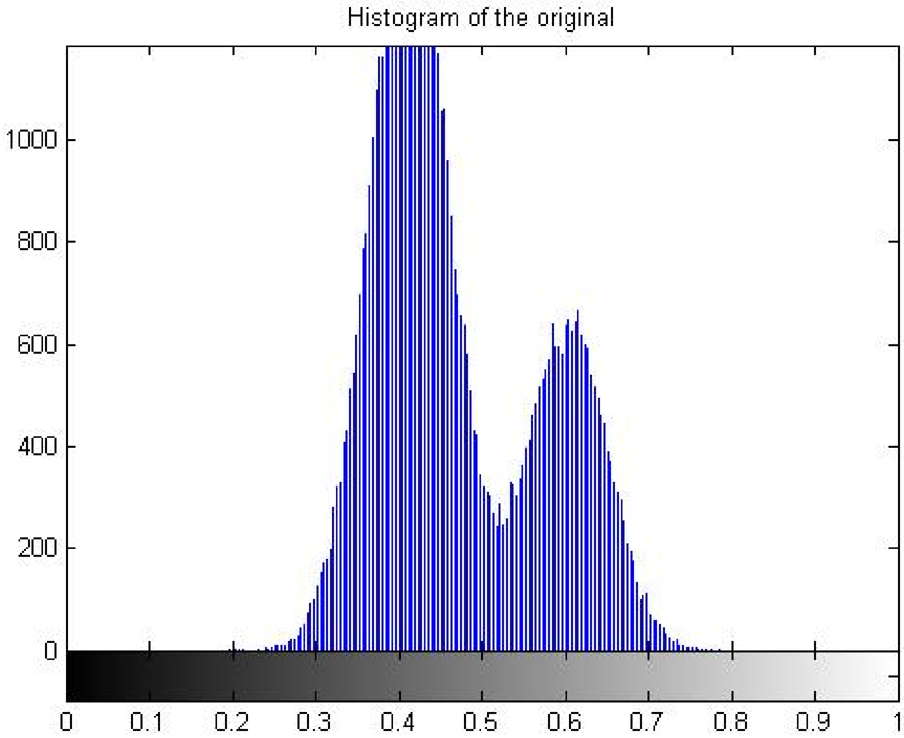
(*a*) Original

(*b*) Nonadaptive

(*c*) Adaptive

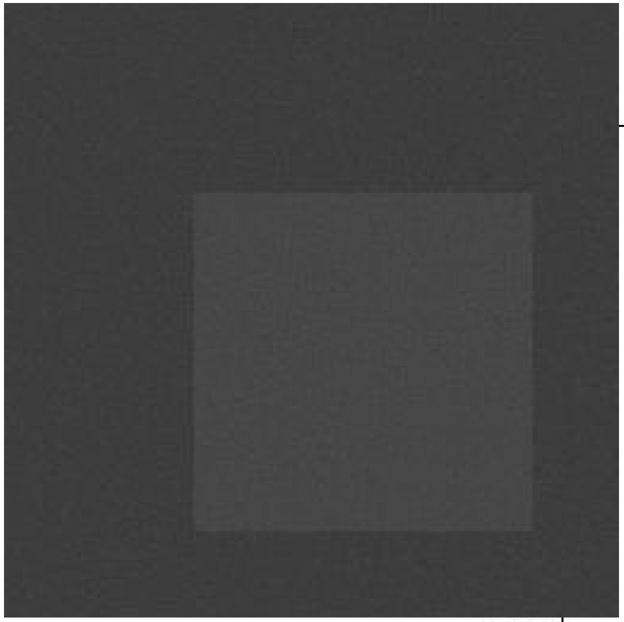
# H. original

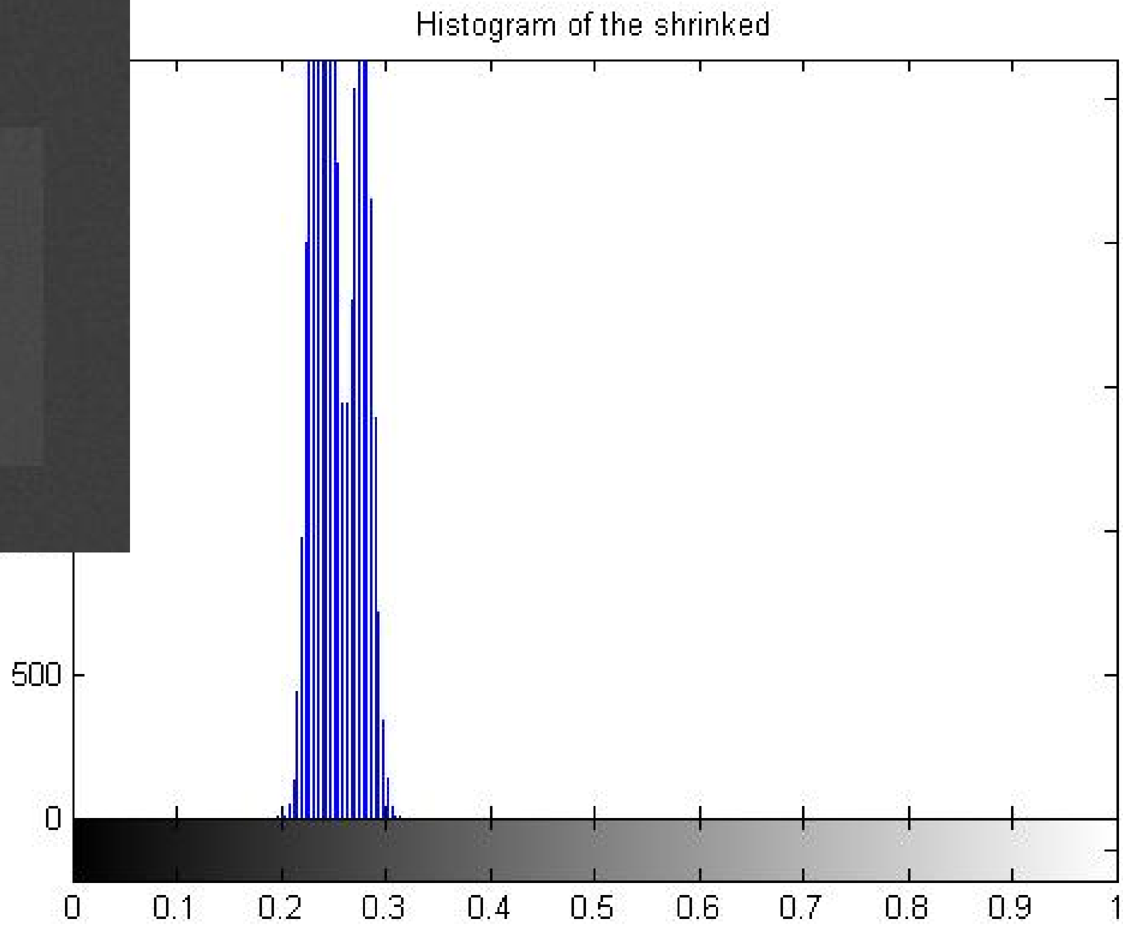




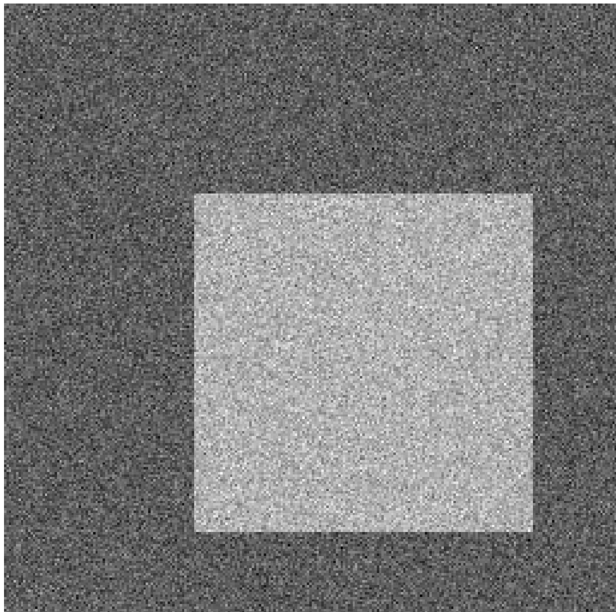
Histogram of the original
1000
800
600
400
200
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

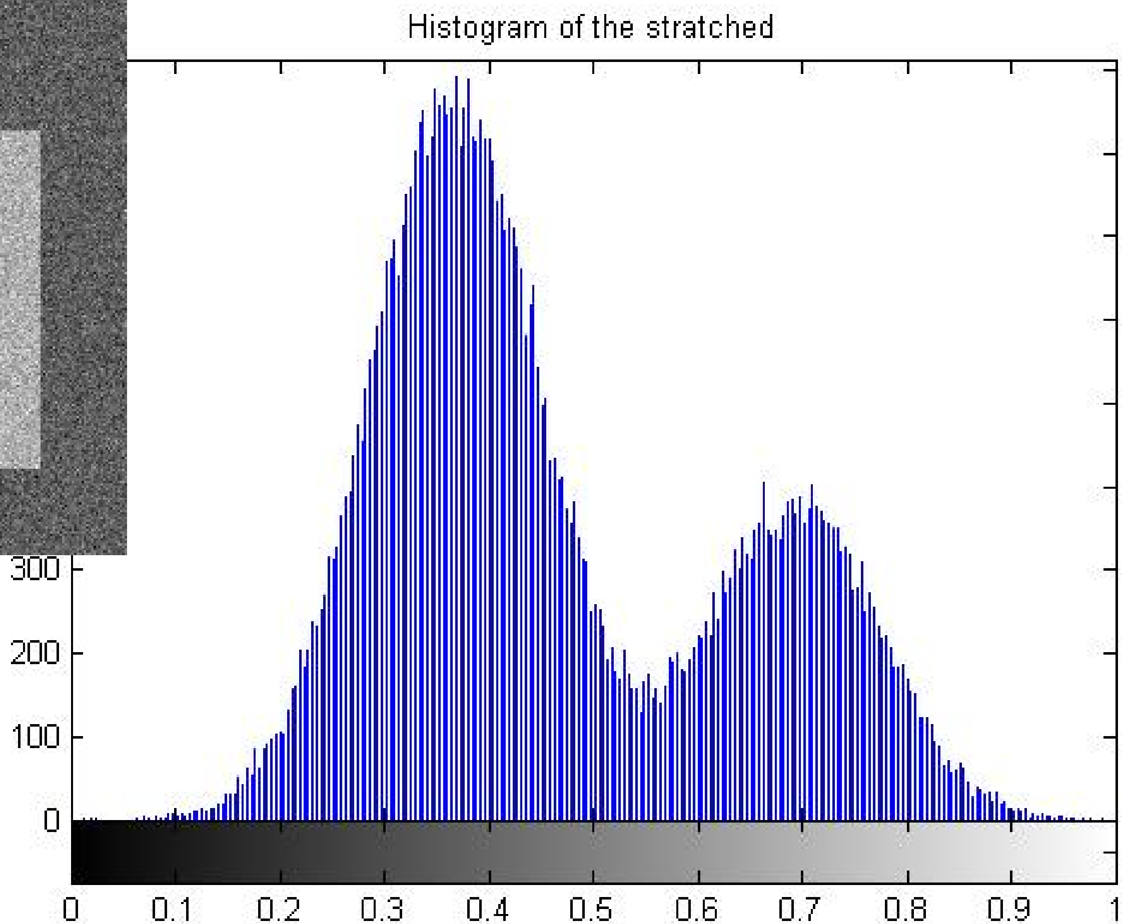
# H. shrinked



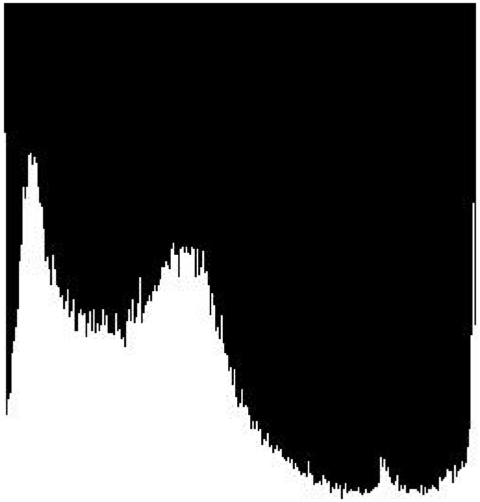


# H. stratched



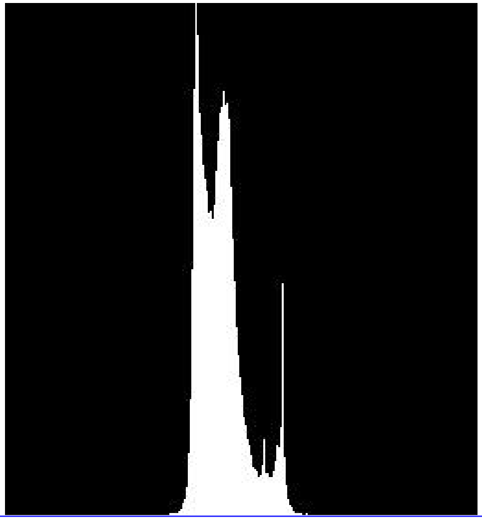

Histogram of the stratched
300
200
100
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

# H. stratching/shrinking

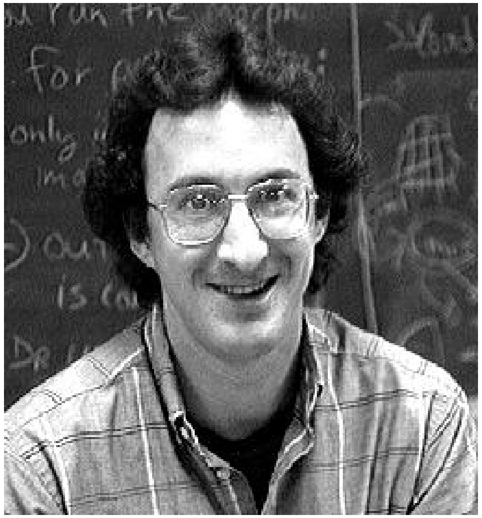


stratching

shrinking



# H. stratching/shrinking



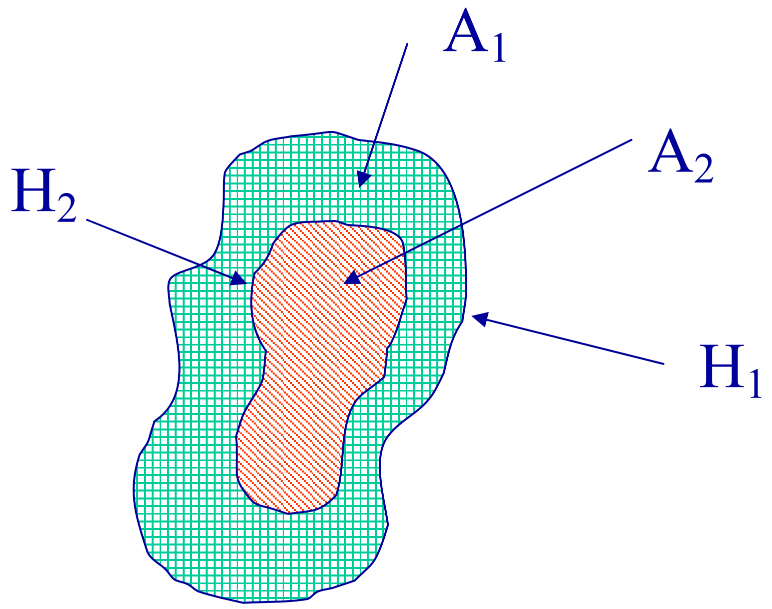






# Example: region-based segmentation

- If the two regions have different graylevel distributions (histograms) then it is possible to split them by exploiting such an information



# Example: region-based segmentation