Image Processing:
Image Enhancement

Michael A. Wirth, Ph.D.
University of Guelph
Computing and Information Science
Image Processing Group

Introduction

- image processing
  - $f(x,y) \rightarrow f'(x,y)$
- image analysis
  - $f(x,y) \rightarrow$ image features
- image understanding
  - $f(x,y) \rightarrow$ high-level image descriptors

Relationships Between Pixels

- A single pixel considered in isolation conveys information on the intensity at a single location in an image
- Perform calculations over regions of an image where the new value of a pixel must be computed from its old value and the values of pixels in its vicinity → neighborhood

Neighbours of a Pixel

- A pixel *p* at coordinates (x,y) has four horizontal and vertical neighbors
  - The coordinates are given by:
    (x+1,y), (x-1,y), (x,y+1), (x,y-1)
  - This set of pixels are called the *4-neighbours*
  - Some of the neighbours of *p* lie outside the image if (x,y) is on the border of the image.
- The four diagonal neighbours of *p* have coordinates:
  - (x+1,y+1), (x+1,y-1), (x-1, y+1), (x-1,y-1)
  - Together with the 4-neighbours, they are called the *8-neighbours* of *p*.

Distance Measures

- For pixels ***p*** and ***q*** with coordinates (x_1,y_1) and (x_2,y_2) respectively:
  - The Euclidean distance between ***p*** and ***q*** is defined as:

$$D_e(p,q) = \left[(x_1 - x_2)^2 + (y_1 - y_2)^2\right]^{\frac{1}{2}}$$

  - Pixels having a distance less than or equal to some value ***r*** from (x_1,y_1) are the points contained in a disk of radius ***r*** centred at (x_1,y_1).

Distance Measures

- The $\mathbf{D_4}$ distance (city-block) between *p* and *q* is defined as:

$$D_4(p,q) = |x_1 - x_2| + |y_1 - y_2|$$

  - The pixels having $\mathbf{D_4}$ distance from (x_1,y_1) less than or equal to some value ***r*** form a diamond centred at (x_1,y_1).

- The $\mathbf{D_8}$ distance (chessboard) between *p* and *q* is defined as:

$$D_8(p,q) = \max\left(|x_1 - x_2|, |y_1 - y_2|\right)$$

Distance Measures

D_4

		2		
	2	1	2	
2	1	0	1	2
	2	1	2	
		2		

D_8

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

On a per pixel basis

- "adding two images together"
  - This means that the addition is carried out between *corresponding* pixels in the two images.

$$g = f_1 + f_2$$

$$g(x, y) = f_1(x, y) + f_2(x, y)$$

Image Enhancement

- The goal of image enhancement is to improve the visual appearance of an image.
  - Visual evaluation of image quality is a highly subjective process
- Approaches to image enhancement fall into two broad categories:
  - *Spatial domain*: direct modification of the pixels in an image.
  - *Frequency domain*: modification of the Fourier transform of an image.

Spatial Domain

- Characterized by *point-by-point* and *neighborhood* transformations

Image Noise

- The process of image acquisition frequently leads (inadvertently) to image degradation.
- Noise is an unexplained variation in intensity values:
  - Noise manifests itself as an unevenness in background and foreground regions
  - Imparts a bumpy or jagged appearance to otherwise smooth regions of intensity

Image Noise

- Spatial noise descriptors:
  - Gaussian noise
  - Rayleigh noise
  - Gamma noise
  - Exponential noise
  - Uniform noise
  - Impulse (salt-and-pepper, shot, spike) noise

Image Noise

Image Noise

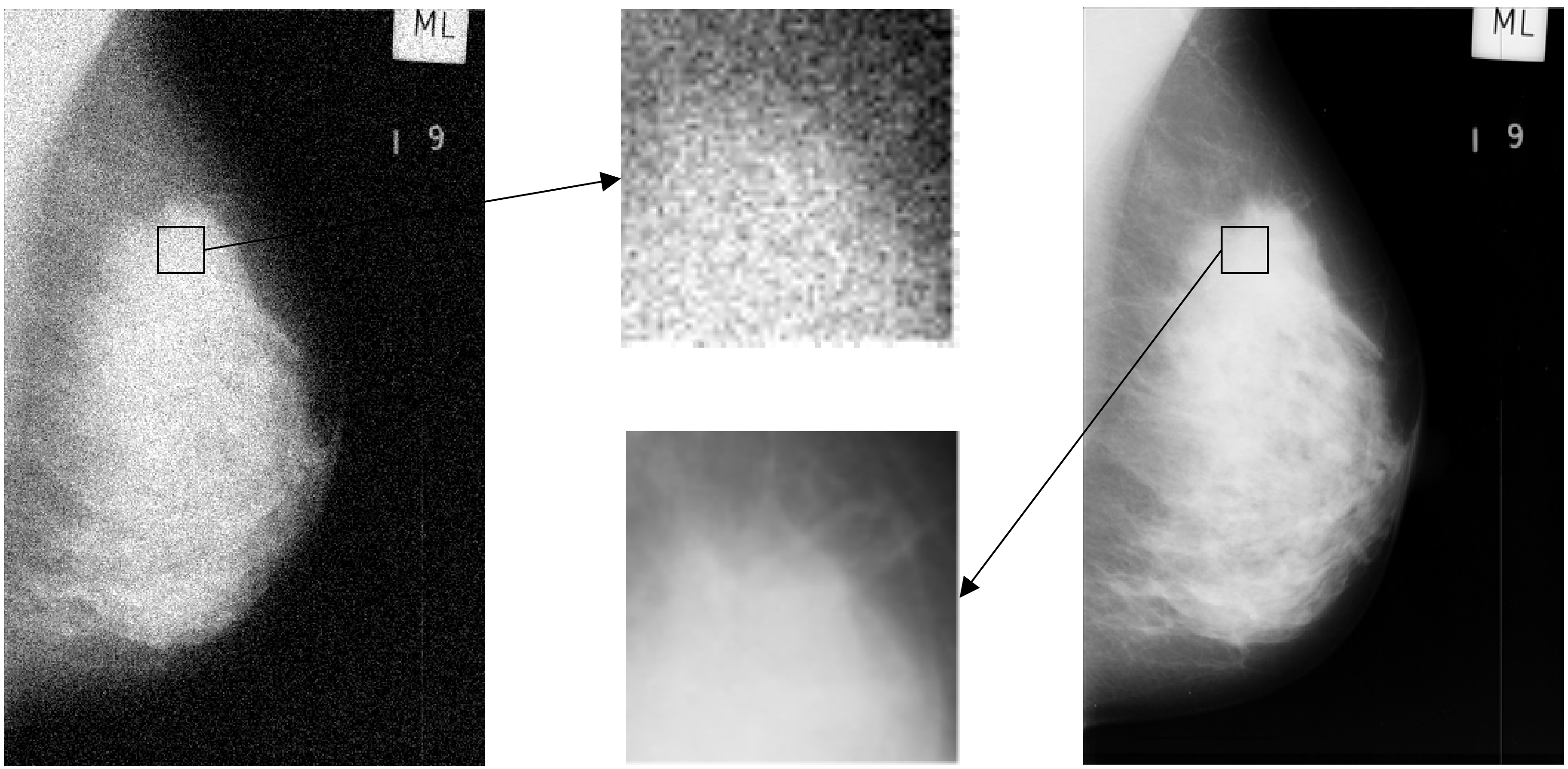


Image Contrast

- For an object to be visible on an image, it is not enough to have sufficient resolution but sufficient contrast also.
- Contrast quantifies the visibility of an object on the background:

$$C = \frac{I_o - I_b}{I_o + I_b}$$

where I_o is the intensity (brightness) of the object and I_b is the intensity (brightness) of the background.

Image Contrast

- Several variations of are frequently used also:

$$C = \left| \frac{I_o - I_b}{I_o + I_b} \right| \qquad C = \frac{I_b - I_o}{I_o + I_b} \qquad C = \frac{I_o - I_b}{I_b} \qquad C = \frac{I_b - I_o}{I_o}$$

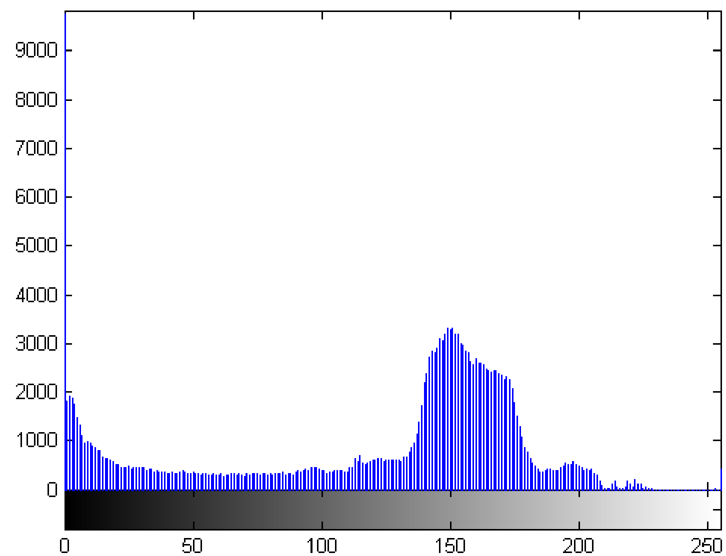
Image Contrast

- Contrast can be clearly illustrated using a histogram
  - Low contrast appears as a narrow set of intensity values, leaving other intensity levels minimally or completely unoccupied.
  - High contrast appears as a broad range of intensity values, with a near uniform distribution.

$$Contrast = \frac{I_{\max} - I_{\min}}{I_{\max} + I_{\min}} \leq 1.0$$

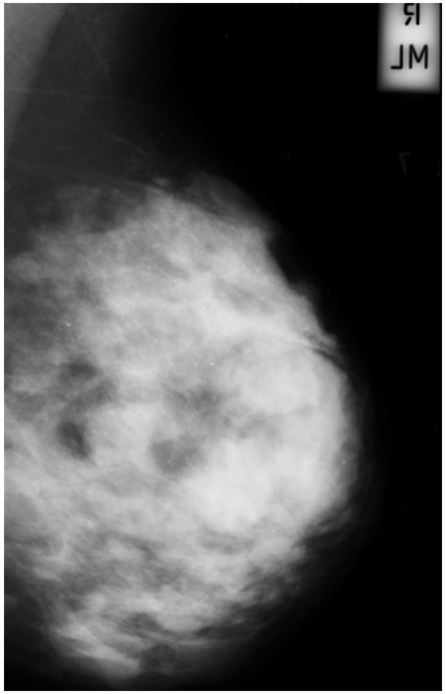
Low-contrast image

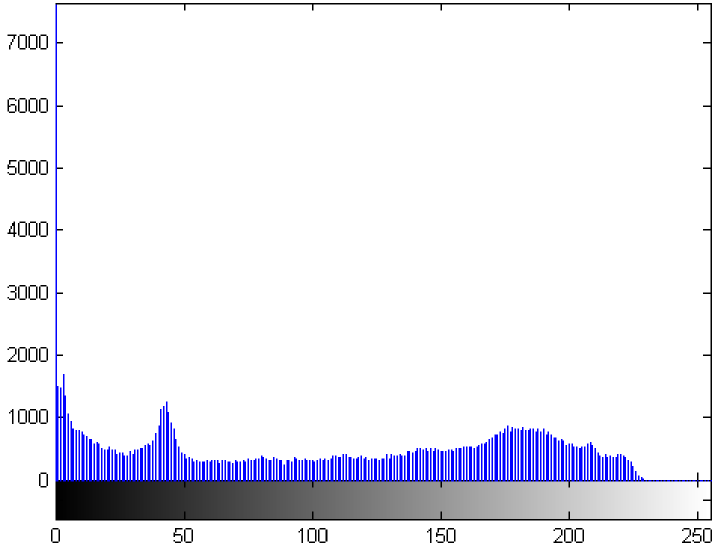
- A histogram that is narrowed and centred towards the middle.



High-contrast image

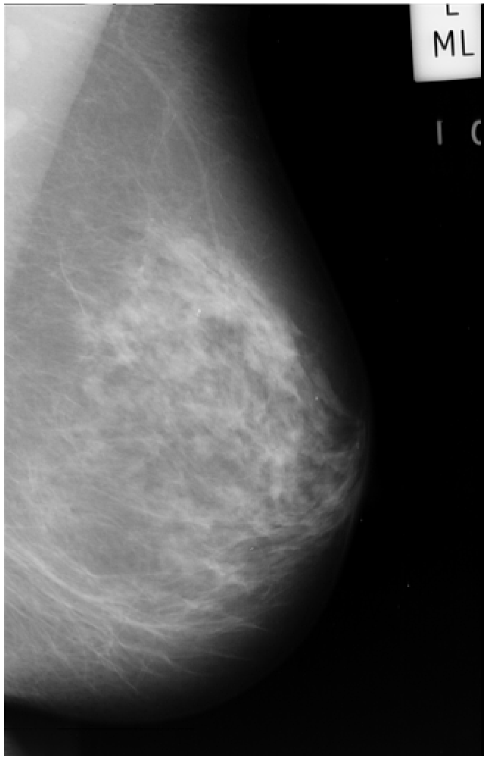
- A histogram that covers a broad range of the intensity values, and a uniform distribution.

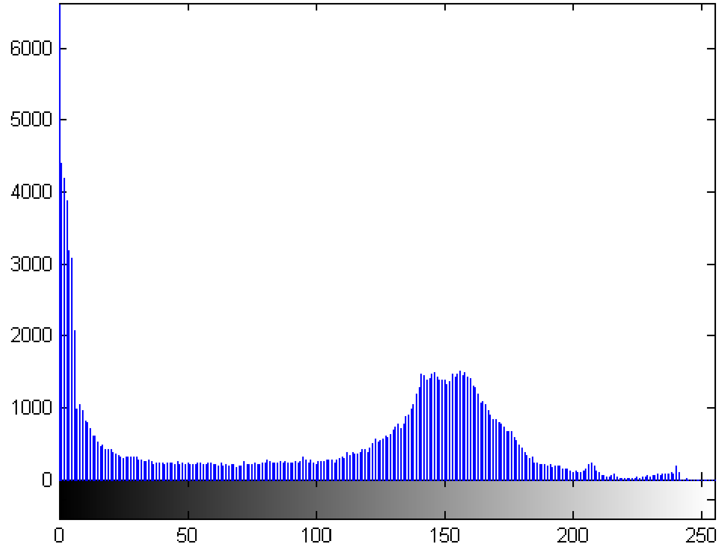




Low-intensity image

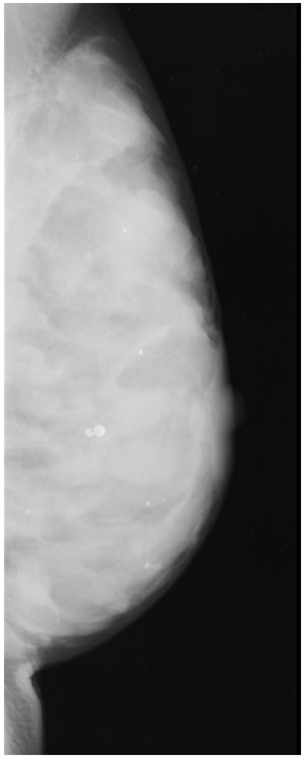
- Biased towards low intensities

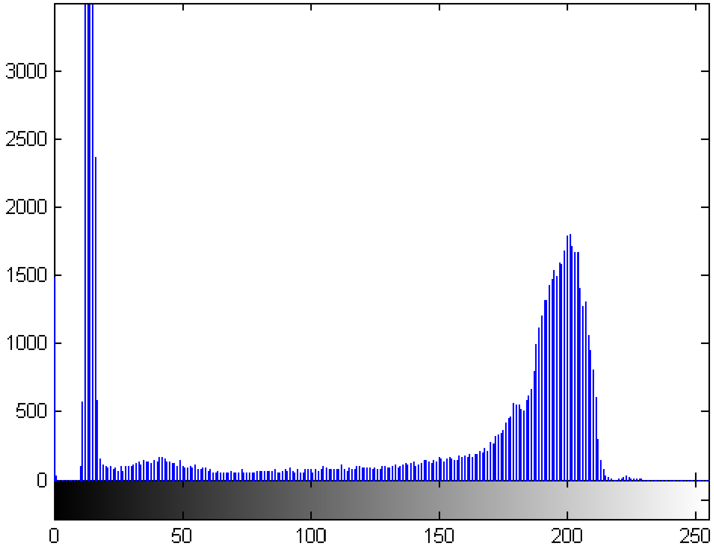




High-intensity image

- Biased towards high intensities





Intensity Transformations

- Spatial transformations are of the form:

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

where *T* is an intensity transformation.

- The simplest form of *T* is when the neighborhood is of size 1×1
  - This is known as point, or point-by-point processing

Point Processing

- There are various methods of pixel manipulation through intensity mapping functions
  - histogram
    - sliding, stretching, equalization
  - linear (negative)
  - logarithmic (log, inverse log, exponential)
  - power-law (n^{th} power and n^{th} root, square-root)

Image Negatives

- The negative of an image with grayscale values in the range [0,L-1] is obtained by using the negative transformation:

$$g(x,y) = (L-1) - f(x,y)$$

Logarithmic

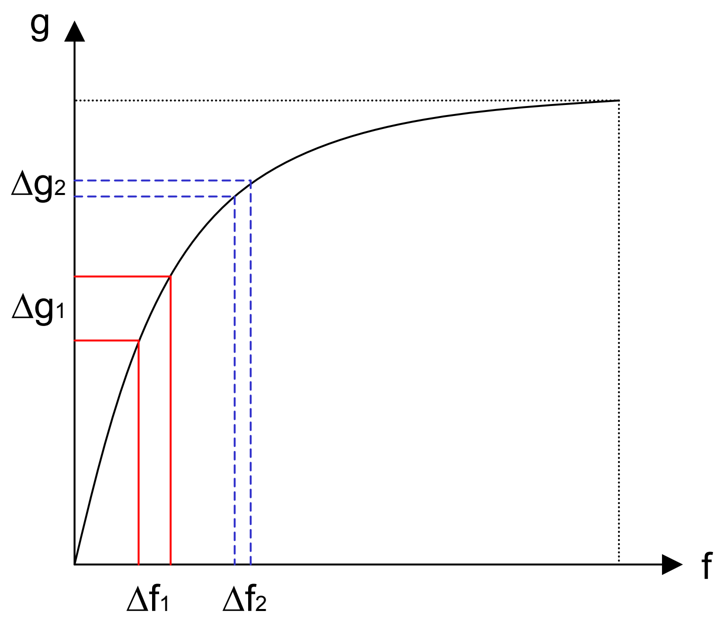
- The general form of the log transformation is:

$$g(x,y) = c\log(1 + f(x,y))$$

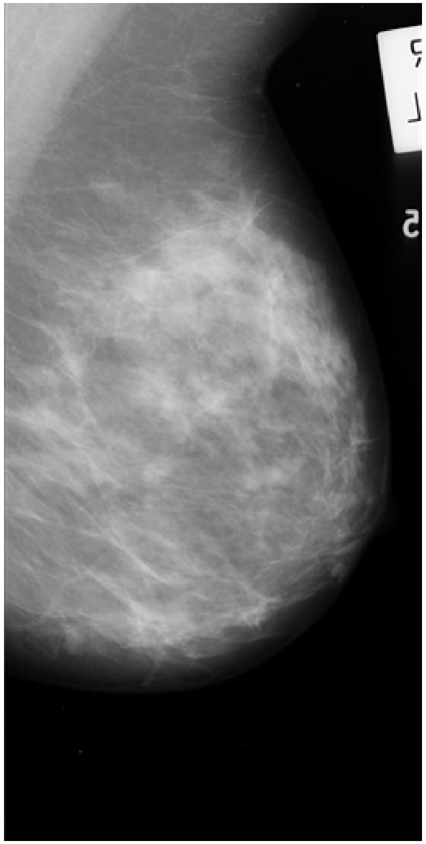
  where c is a constant, and it is assumed that $f(x,y) \geq 0$.
- This transformation maps a narrow range of low grayscale values in the input image into a wider range of output values.

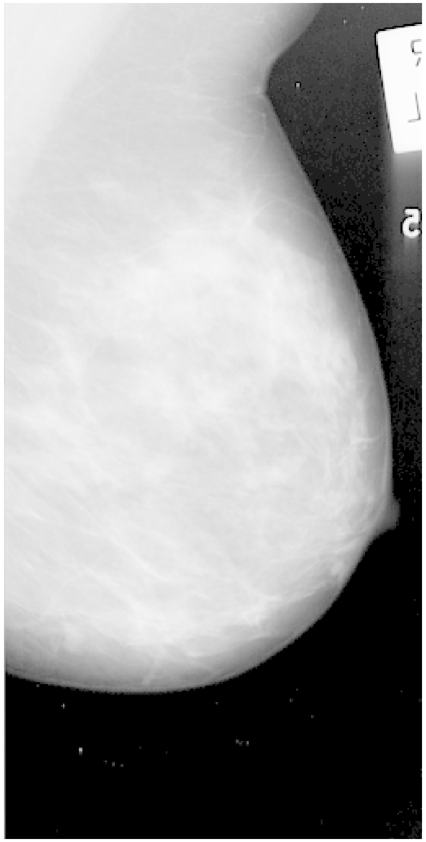
Logarithmic

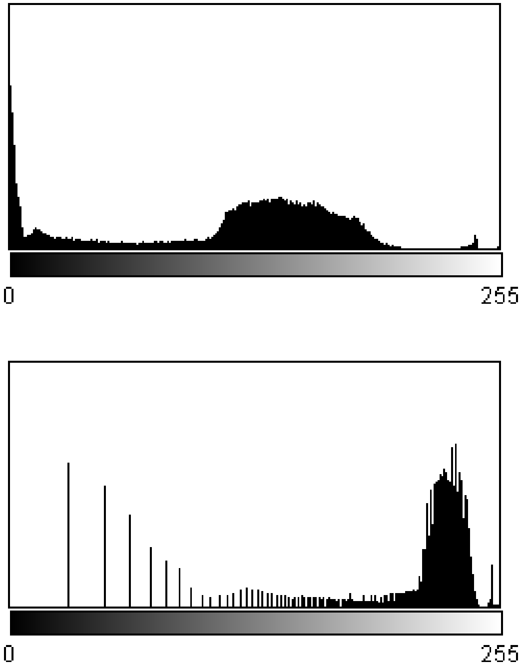
- It expands the values of low-intensity pixels, compressing higher-level intensities.
  - $\Delta f_1 < \Delta g_1$ contrast is increased
  - $\Delta f_2 > \Delta g_2$, contrast is reduced



Logarithmic





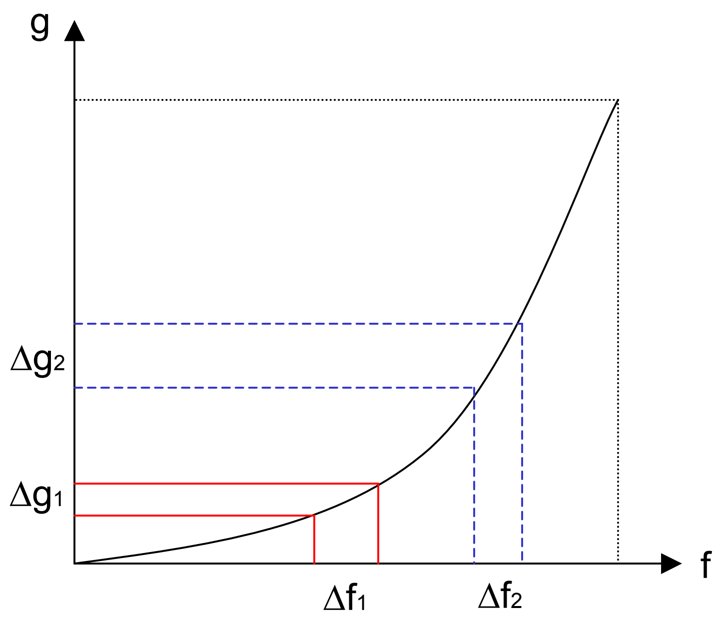


Inverse Logarithmic

- The inverse logarithmic transformation compresses the dynamic range of images with large variations in pixel values.

Exponential

- Uses a exponential function to perform the intensity mapping.
  - Enhances low-intensity regions
  - $\Delta f_1 > \Delta g_1$ contrast is reduced
  - $\Delta f_2 < \Delta g_2$, contrast is increased



Power-law

- Power-law transformations have the following form:

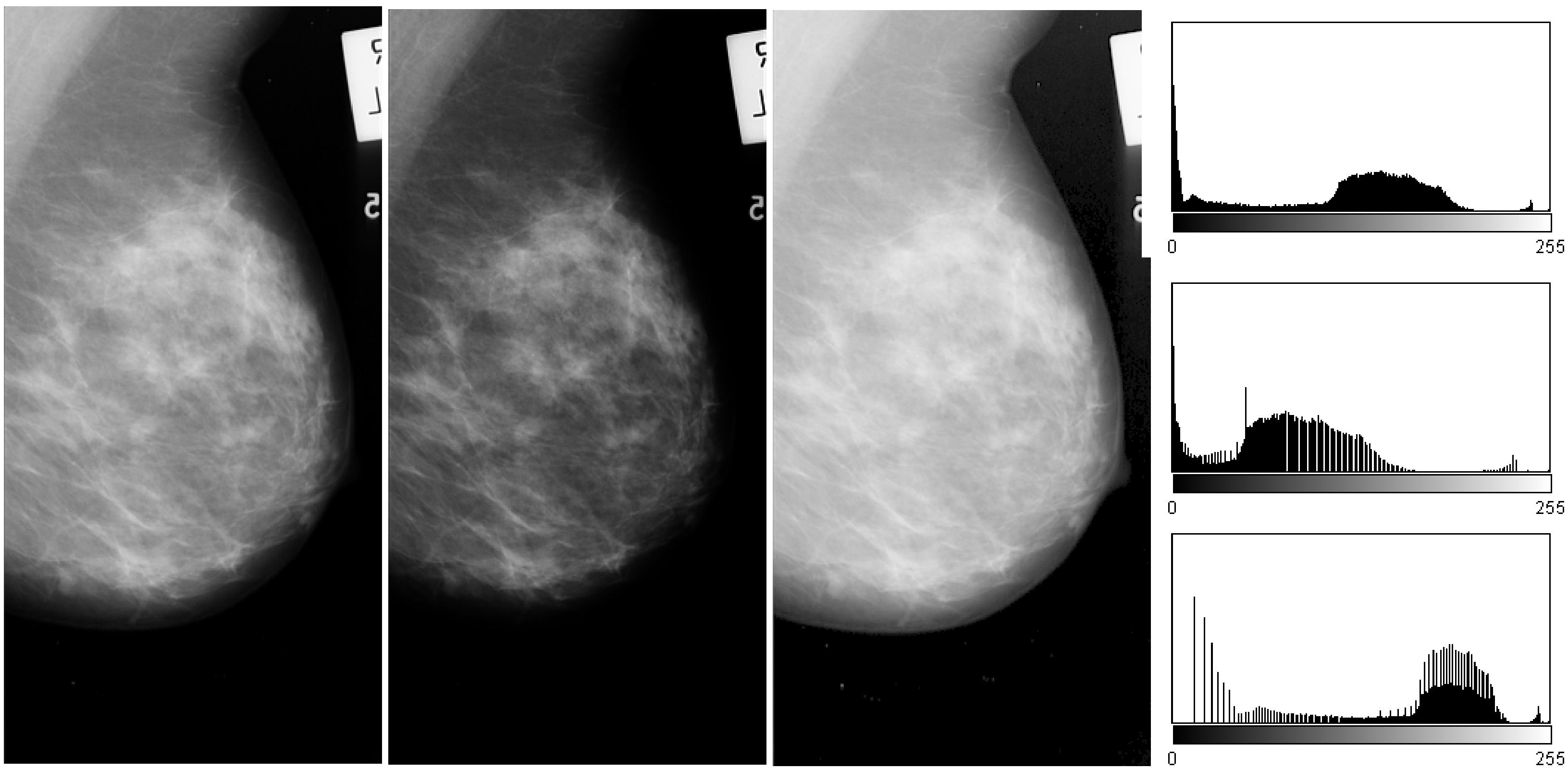
$$g(x,y) = cf(x,y)^{\gamma}$$

where *c* and γ are positive constants

e.g. square-root γ *= 0.5*

Power-law

Squared versus SQRT



Linear vs. Nonlinear Transformations

- A transformation T, is said to be linear if, for any two images *f* and *g* and two scalars *a* and *b*:

$$T(af + bg) = aT(f) + bT(g)$$

- e.g. An operation to compute the sum of K images is *linear*
- e.g. An operation to compute the absolute value of the difference of two images is not → *nonlinear*

Local vs. Global Transformations

- A *global* transformation is performed using the entire image.
- A *local* transformation is performed by dividing the image into sub-images and processing each of these independently.
  - Also called *adaptive* or *variable*

Histogram

- A histogram represents the frequency distribution of intensity values in an image.
  - For example a histogram of an 8-bit image has 256 “bins” indexed from 0 to 255

Histogram

- The histogram of an image with grayscale values in the range [0,L-1] is a function of the form:

$$h(r_k) = n_k$$

where r_k is the k^{th} intensity value and n_k is the number of pixels in the image having intensity value k.

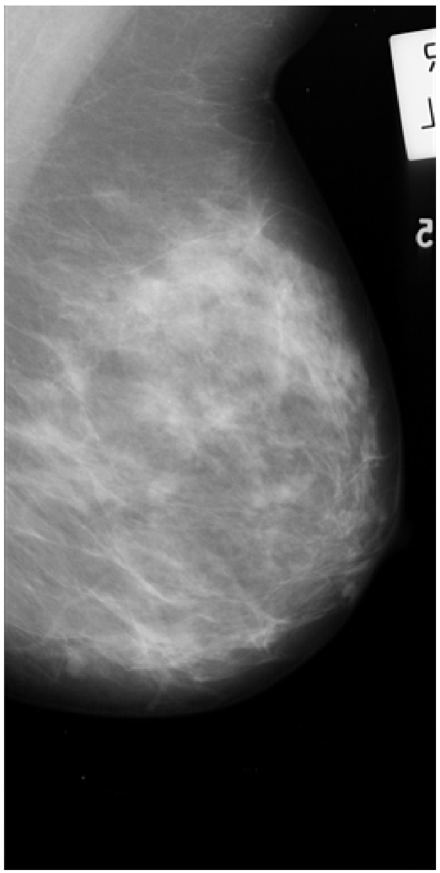
Histogram Normalization

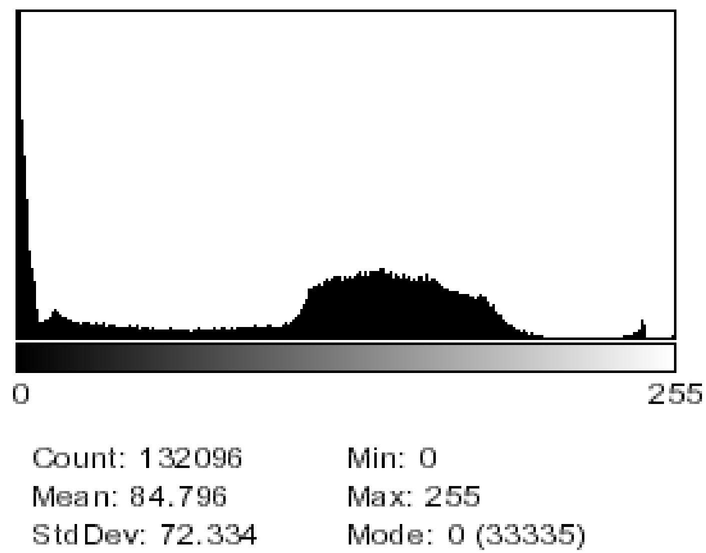
- A histogram can be normalized by dividing each of its values by the total number of pixels in the image, denoted by *n*.

$$p(k) = h_k / n$$

  p(k) gives an estimate of the probability of occurrence of intensity value *k*.
- The sum of all the bins in a normalized histogram is equal to 1.

Histogram




0
255
Count: 132096
Mean: 84.796
StdDev: 72.334
Min: 0
Max: 255
Mode: 0 (33335)

ALGORITHM: Calculation of an image histogram

Create an array *histogram* with 2^b elements

```
for i = 1:max_intensity
   histogram[i] = 0;
end
for x = 1:nrows
    for y = 1:ncolumns
        D = image[x,y];
        histogram[D] = histogram[D] + 1;
    end
end
```

Histogram Descriptors

- A number of global image statistical parameters can be derived from the histogram.
- Maximum, Minimum
  - The pixels with the highest and lowest intensity values.
- Mean
  - The average intensity value of the image.

$$m = \frac{\sum_{i=0}^{L-1} h(i)}{n}$$

Histogram Descriptors

- Variance
  - A measure of the width of the histogram.

$$v = \sum_{i=0}^{L-1} (h(i) - m)^2$$

- Skewness
  - A measure of the non-symmetric distribution of the histogram.

$$sk = \sum_{i=0}^{L-1} (h(i) - m)^3$$

Histogram Descriptors

- Kurtosis
  - A measure of the deviation of the histogram from a Gaussian profile.

$$k = \sum_{i=0}^{L-1} (h(i) - m)^4 - 3$$

- Entropy
  - A measure of “choas” or noise in the image.

$$e = \sum_{i=0}^{L-1} \left\{ \frac{h(i)}{(n^2 \times m)} \cdot \log_2 \left(\frac{h(i)}{n^2 \times m} \right) \right\}$$

Histogram Descriptors

- Standard Deviation
  - A measure

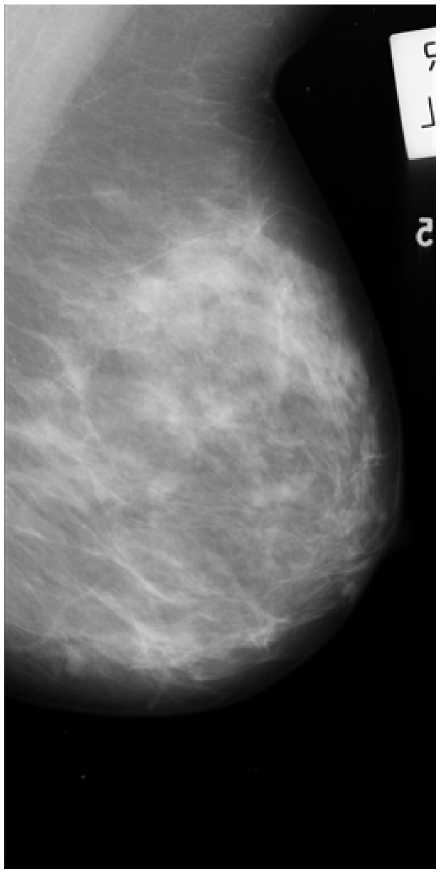
$$sd = \sqrt{v}$$

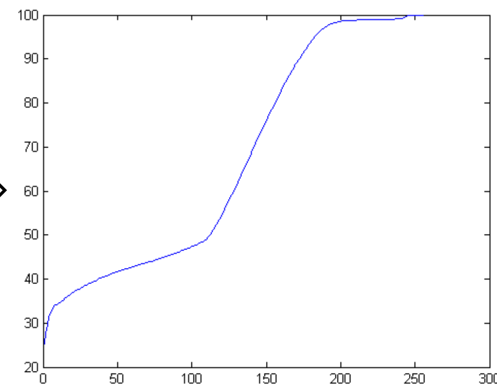
Cumulative Histogram

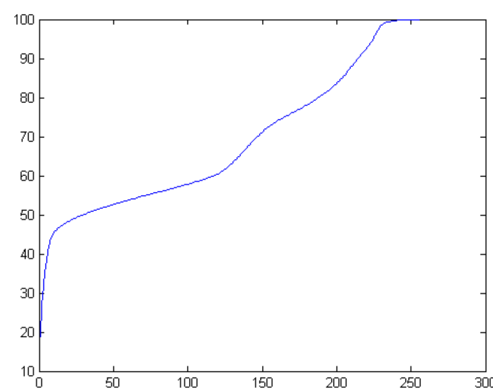
- A cumulative histogram represents the cumulative frequency distribution of intensity values in an image.
  - The cumulative frequency of an intensity value, *i*, is the number of times that an intensity value less than or equal to *i* occurs in an image.
  - Cumulative frequencies, c_j, are computed from histogram counts, $h(i)$:

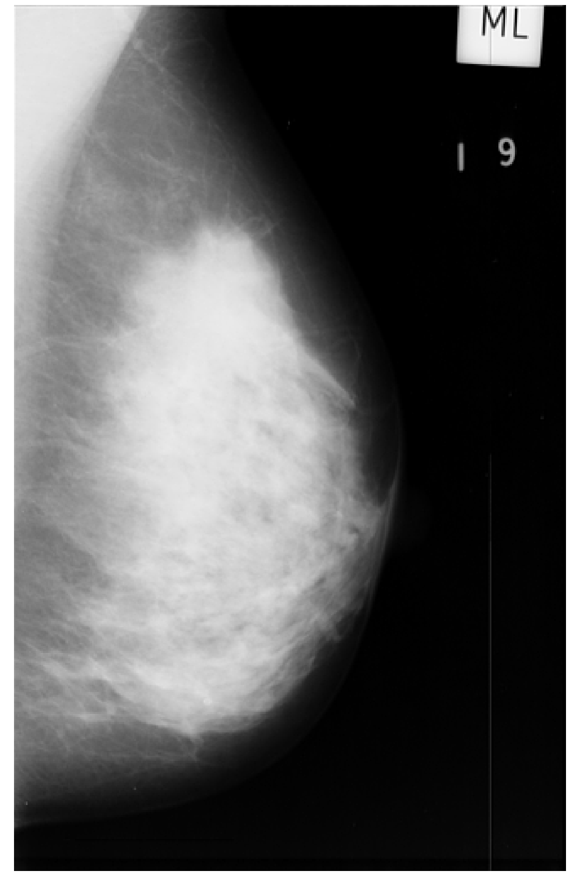
$$c_j = \sum_{i=0}^{j} h(i) \qquad j = 0, 1, \ldots, L-1$$

Cumulative Histogram







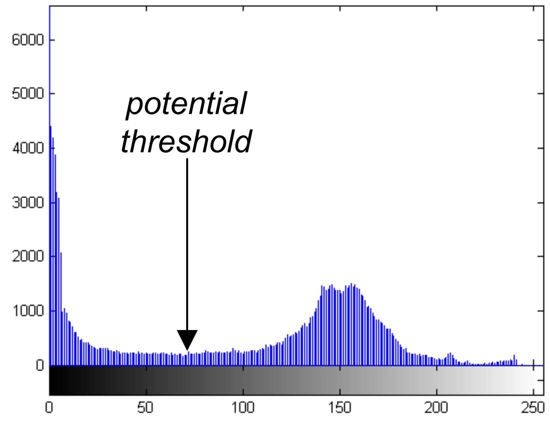


ALGORITHM: Calculation of a cumulative histogram

```
Create an array histogram with 2^b elements

sum = 0;
for i = 1:max_intensity
    sum = sum + histogram[i];
    cumulative_histogram[i] = sum;
end
```

Histograms

- One of the principal uses of the histogram is the selection of threshold parameters.
- It is useful to plot *h(i)* as a function of *i*.
  - From this graph a suitable position for the threshold can be related directly to the position of the *foot of a hill* or *valley* in the histogram.



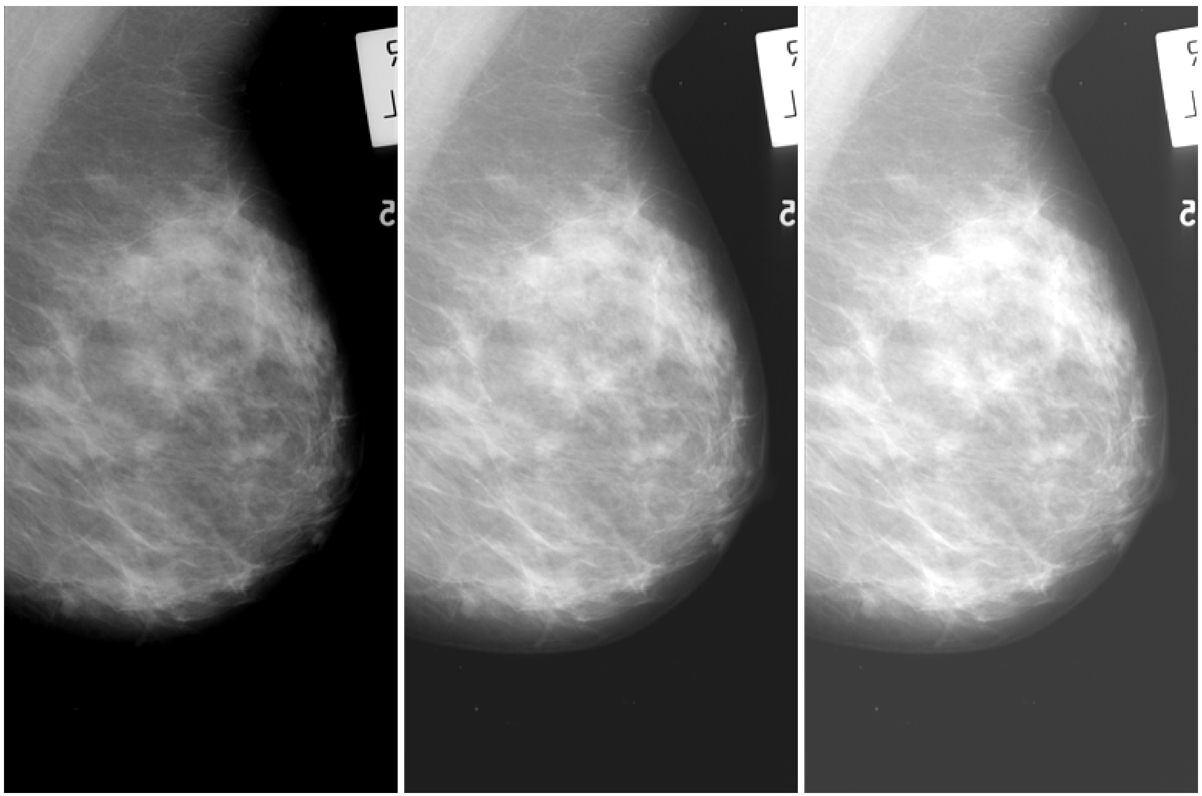
Histogram Sliding

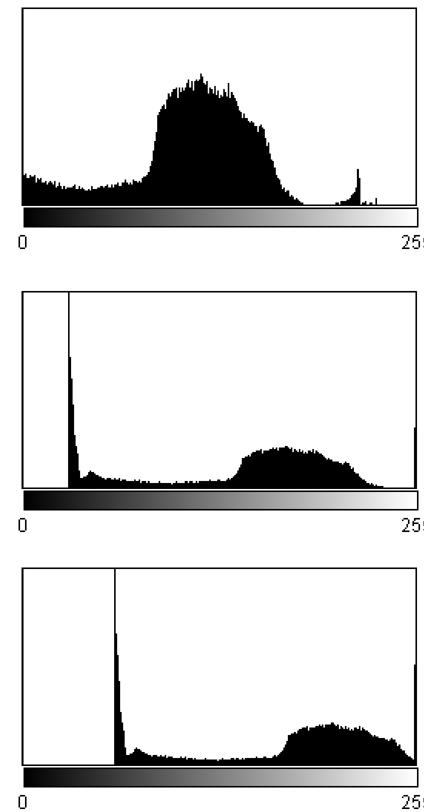
- Histogram sliding involves the addition or subtraction of a constant value, *b*, to all the pixels in an image.
  - It is sometimes referred to as adding *bias* to the image intensity

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

  - If $b > 0$, then brightness is increased; if $b < 0$, it is decreased.

Histogram Sliding





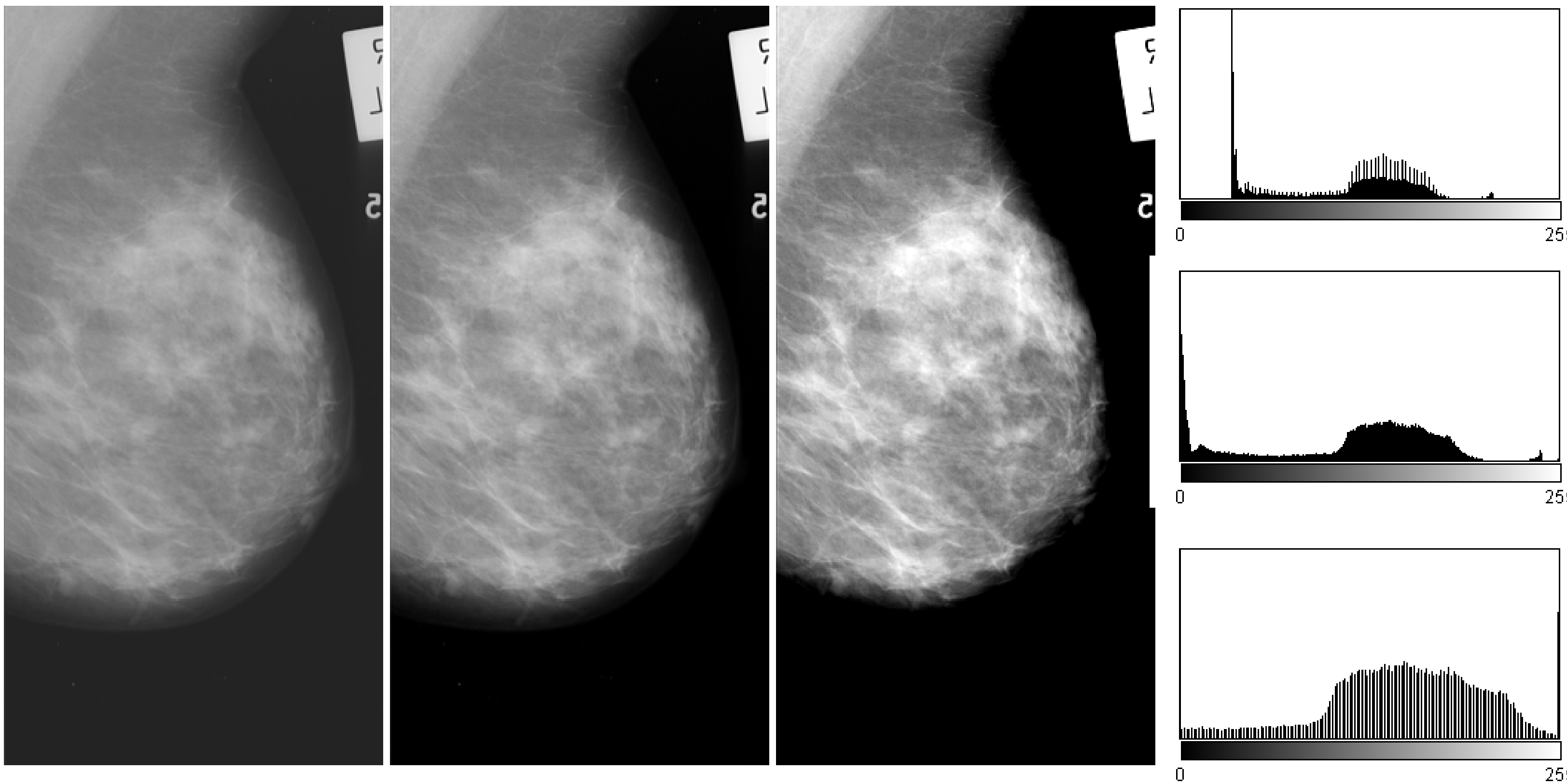
Histogram Stretching

- Histogram stretching involves the multiplication or division of all pixels in an image by a constant value, *a*.
  - It is sometimes referred to as adding *gain* to the image intensity

$$g(x,y) = af(x,y)$$

  - If *a* > 1, then contrast is increased; if *a* < 1, it is reduced.

Histogram Stretching



Histogram Sliding & Stretching

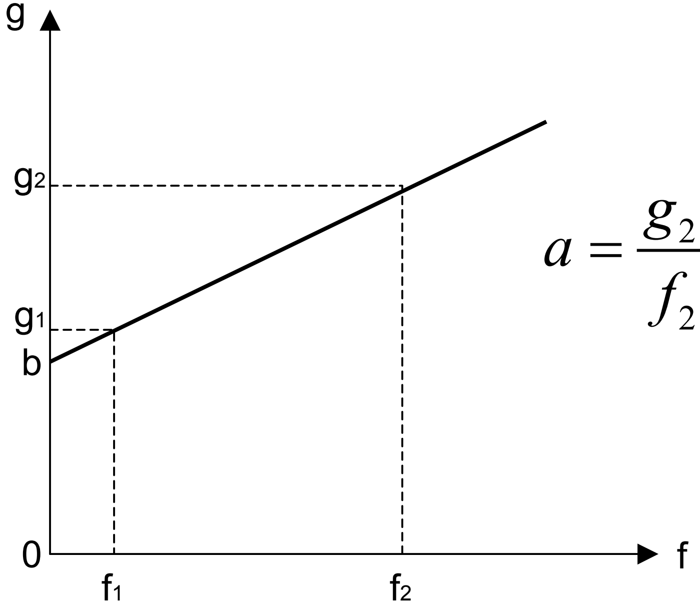
- Histogram stretching and sliding can be combined to give a general expression for brightness and contrast modification:

$$g(x,y) = af(x,y) + b$$

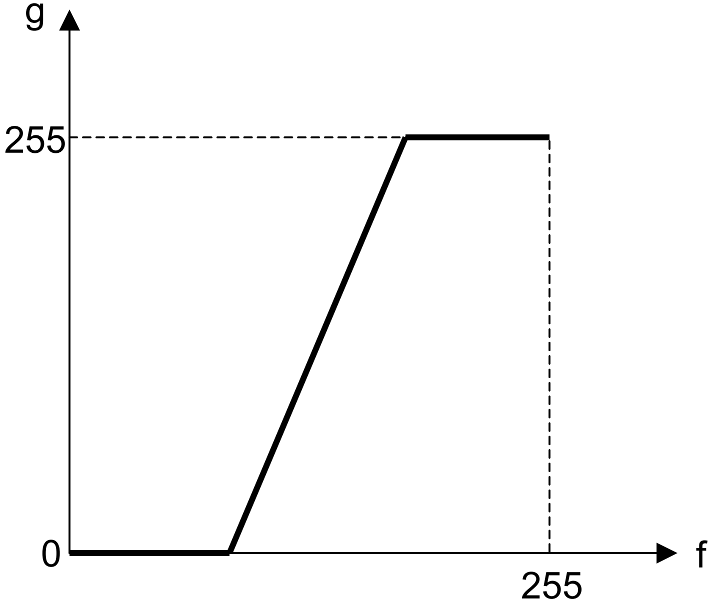
- To map a particular range of intensity values $[f_1,f_2]$ onto a new range $[g_1,g_2]$

$$g(x,y) = g_1 + \left(\frac{g_2 - g_1}{f_2 - f_1}\right)\left[f(x,y) - f_1\right]$$

Histogram Sliding & Stretching



$$a = \frac{g_2 - g_1}{f_2 - f_1}$$



Linear vs. Nonlinear Mappings

- In linear mapping the gain, *a*, is static
- In nonlinear mapping the gain, *a*, can vary.
  - The way in which contrast is modified depends on the input intensity value

    e.g. logarithmic, exponential functions

Histogram Equalization

- Histogram equalization or linearization redistributes intensity values in an attempt to "flatten" the frequency distribution.
  - Involves normalizing the histogram
  - For each intensity level *j* in the original image histogram, the new intensity value *K* is calculated as:

$$k_j = \sum_{i=0}^{j} \frac{h(i)}{T}$$

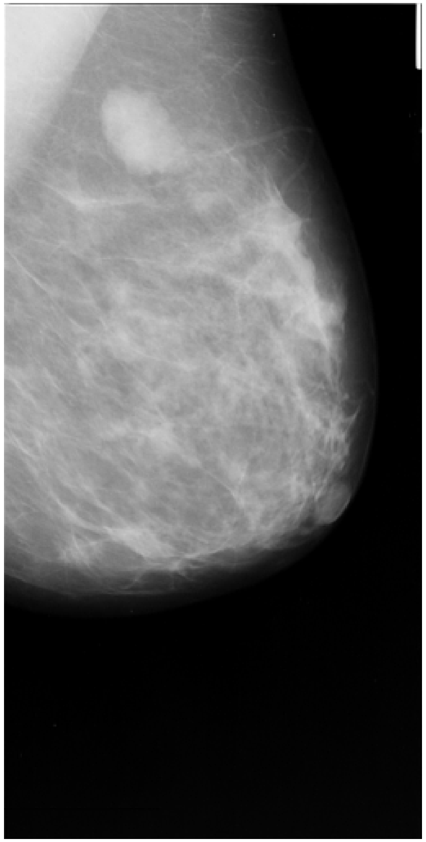
h(i) is the number of pixels with intensity value *i*.

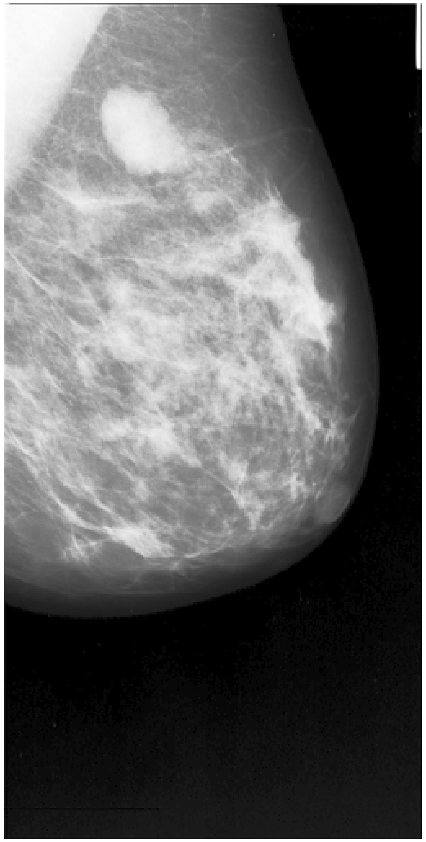
T is the total number of pixels in the image and j = 0,1,2,…,L-1

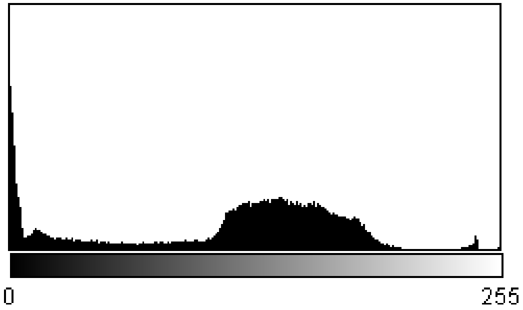
ALGORITHM: Histogram equalization

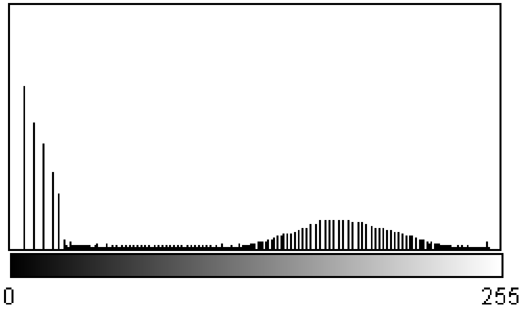
```
Compute a scaling factor, b=255/number of pixels
Calculate histogram

c[1] = b * histogram[1];
for i = 2:max_intensity
    c[i] = c[i-1] + b*histogram[i];
end
for x = 1:nrows
    for y = 1:ncolumns
        D = image[x,y];
        g[x,y] = c[D+1];
    end
end
```

Histogram Equalization









Histogram Equalization

Histogram Specification

- A method which generates an image that has a *specified* histogram.

Convolution

- One of the fundamental operations in image processing
  - Enhancement techniques based on this type of approach are often referred to as *spatial filtering*.

Convolution

- In convolution, the calculation performed at a pixel is a weighted sum of intensity values from a neighborhood surrounding a pixel.
  - If a neighborhood is centred on a pixel then it must have odd dimensions
  - Intensity values from a neighborhood are weighted by coefficients that come from a convolution kernel, or mask

Convolution

- The kernel is usually small relative to the image

    e.g. 3x3 is most common

$i-1,j-1$	$i-1,j$	$i-1,j+1$
$i,j-1$	i,j	$i,j+1$
$i+1,j-1$	$i+1,j$	$i+1,j+1$

Convolution

- During convolution, each kernel coefficient is taken in turn and multiplied by a value from the neighborhood of the image lying under the kernel.

$$g(x,y) = \sum_{j=-1}^{1} \sum_{i=-1}^{1} w(i,j) f(x-i, y-j)$$

- For example:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Convolution

- For a kernel of width *m* and height *n*, both odd:

$$g(x,y) = \sum_{j=-n2}^{n2} \sum_{i=-m2}^{m2} w(i,j) f(x-i, y-j)$$

the kernel half-width m_2, and half-height n_2 are given as:

$$m2 = \lfloor m/2 \rfloor, \qquad n2 = \lfloor n/2 \rfloor$$

Kernels

- An omnidirectional kernel is one whose response is the same, whatever the direction in which intensity values vary.
- A uniform kernel has coefficients which all have the same weight.
- A nonuniform kernel has coefficients which have differing weights.

Kernel Shapes

- Rectangular versus circular (pill-box) uniform (smoothing) kernels

$$w(i,j)=\frac{1}{25}\begin{bmatrix}1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\end{bmatrix} \qquad w(i,j)=\frac{1}{21}\begin{bmatrix}0&1&1&1&0\\1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\\0&1&1&1&0\end{bmatrix}$$

Kernel Shapes

- Pyramidal versus cone nonuniform (smoothing) kernels

$$w(i,j)=\frac{1}{81}\begin{bmatrix}1&2&3&2&1\\2&4&6&4&2\\3&6&9&6&3\\2&4&6&4&2\\1&2&3&2&1\end{bmatrix}\quad w(i,j)=\frac{1}{25}\begin{bmatrix}0&0&1&0&0\\0&2&2&2&0\\1&2&5&2&1\\0&2&2&2&0\\0&0&1&0&0\end{bmatrix}$$

Separable Kernels

- A separable $n \times n$ kernel is represented as a vector product of two orthogonal 1D kernels, each of width n.
  - One is applied down the columns of an image, generating an intermediate result.
  - The other kernel is then applied along the rows of the intermediate image, producing the final result.

The Convolution Process

1. Place the centre of the kernel over a pixel, *P*, in the input image.
2. Multiply each pixel in the $k \times k$ neighborhood by the appropriate filter kernel coefficient superimposed on it.
3. Sum all the products.
4. Place the suitably normalized sum into the pixel position, *P*, in the output image

ALGORITHM: Image convolution

```
Create a kernel, w, with dimensions m × n
Compute kernel half-width, m2
Compute kernel half-height, n2
Create an M × N output image, g

for x = 1:nrows
    for y = 1:ncolumns
        g[x,y] = 0;
    end
end
```

ALGORITHM: Image convolution

```
for x = m2:M-m2
    for y = n2:N-n2
        sum = 0;
        for i = -m2:m2
            for j = -n2:n2
                sum = sum+w[i+m2,j+n2]*image[x-i,y-j];
            end
        end
        g[x,y] = sum;
    end
end
```

Image Borders

- Along the borders of an image it is not possible to compute a convolution, because part of the kernel lies beyond the image.
  - True of any neighborhood operation
- The size of the region in which normal convolution is possible is dictated by the dimensions of the convolution kernel.
  - e.g. for a 3x3 kernel: 1-edge border

Image Borders

- For certain values of *x* and *y*, one or both of the expressions (*x*-*i*) and (*y*-*j*) will give a value outside the allowed range [0,M-1][0,N-1]

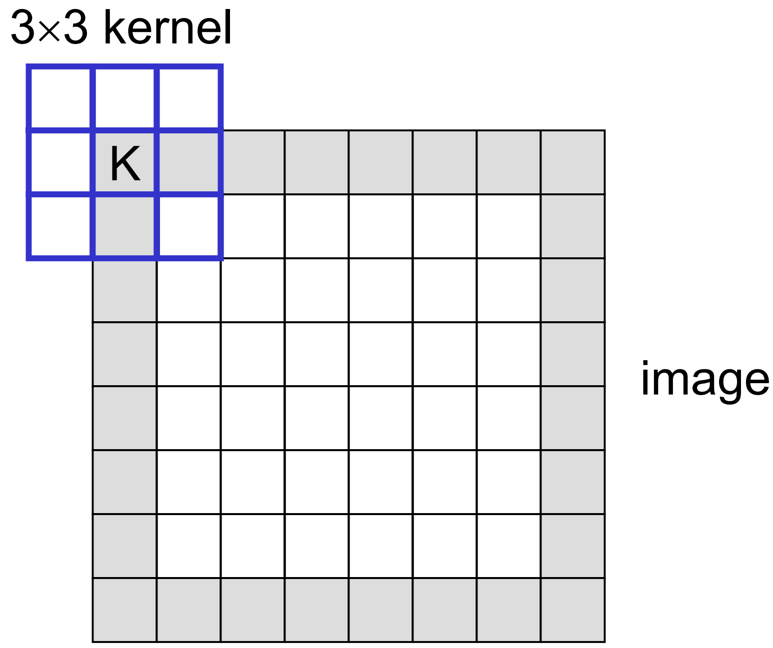


Image Borders

- For an image $f(x, y)_{M \times N}$ the region to which the kernel applies is (M-2) $\times$ (N-2) with an origin at (1,1)
- In general an origin at (m_2,n_2) and dimensions of (M-2m_2) $\times$ (N-2n_2)
- A number of different strategies exist to deal with this problem

Image Borders

1. No processing at the border
   - Ignore those pixels for which convolution is not possible
2. Copy of input pixels
   - Copy the corresponding pixel value from the input image wherever it is not possible to carry out convolution.
   - The image will have a border of unprocessed pixels

Image Borders

3. Truncation of the image
   – Remove those pixels for which convolution is not possible.
   – The resulting image is smaller than, and offset to, the input image.

Image Borders

4. Truncation of the kernel
   - Deal with the borders of the image as a special case and use a *modified kernel* to perform convolution.

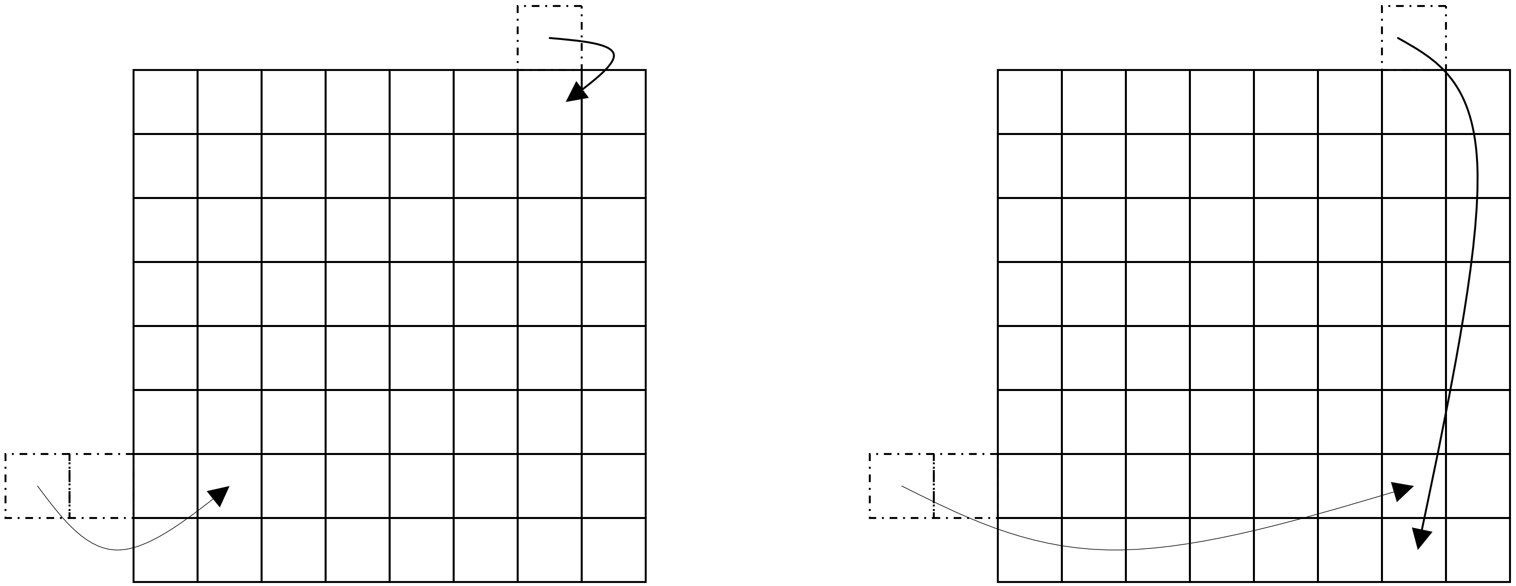
$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \qquad \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}$$

   - Adds considerably to the complexity of convolution.

Image Borders

5. Reflected indexing
   – Test (*x-i*) to see whether it corresponds to a valid pixel value, and if it doesn't, the coordinate can be reflected-back into the image.
   – The same can be done at (*y-j*)
   – Simulates mirroring of the image at its borders
6. Circular indexing
   – Imaging that the image repeats itself endlessly in all directions.

Image Borders



Reflected and Circular indexing

Spatial Frequency

- Spatial frequency is a measure of how rapidly intensity varies as an image is traversed.
  - Images in which intensity varies slowly and smoothly → low spatial frequency
  - Images with sudden intensity transitions, fine detail and strong texture → high spatial frequency

Linear Filtering

- Convolution can be used to carry out linear filtering of an image.
  - The response is given by a sum of products of the kernel coefficients and the corresponding image pixels in the area spanned by the filter mask.
  - There are two complementary types of linear filters: spatial or frequency filters.

Image Smoothing

- Image smoothing or low-pass filtering, allows low spatial frequencies to remain unchanged, but suppresses high frequencies.
  - A low-pass filter has the effect of smoothing or blurring the image, reducing noise but obscuring fine detail.

Image Smoothing

- The filter replaces the value of every pixel in an image by the average of the intensity values in the neighborhood defined by the kernel.
  - The resulting image has reduced “sharp” transitions in the intensity values.
  - Noise consists of sharp transitions → noise reduction
  - Edges are also characterized by sharp transitions in intensities, so smoothing filters blur edges.

Image Smoothing

- Any convolution kernel whose coefficients are all positive will act as a low-pass filter.
- In the simplest case, all coefficients are equal.

$$w = \begin{bmatrix} 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \end{bmatrix}$$

- Note that the kernel has already been normalized. It’s coefficients sum to 1.

Image Smoothing

- Factor out the normalization:

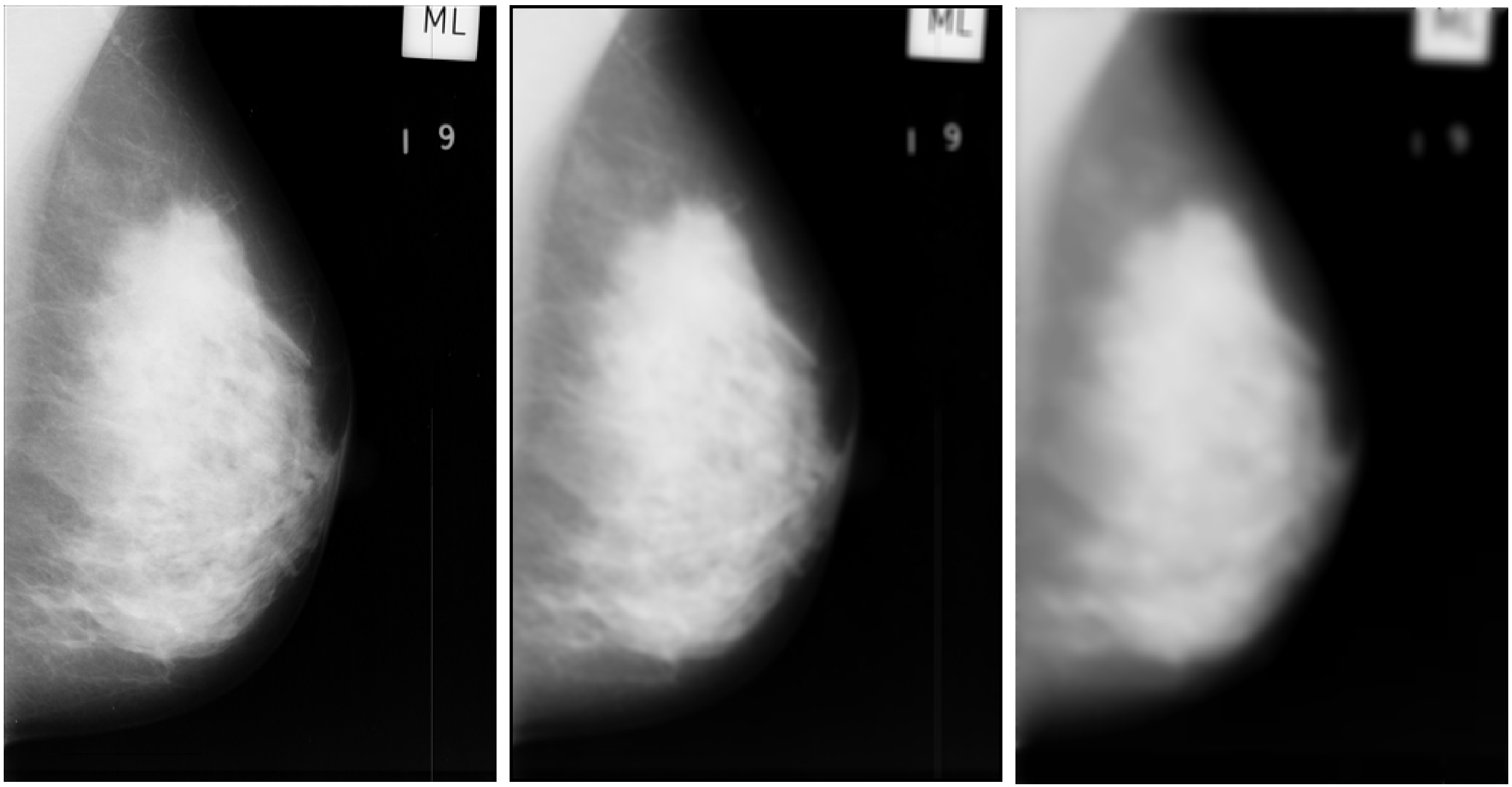
$$w(i,j) = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

- Pixel values in the neighborhood are summed without being weighted, and the sum is divided by the number of pixels in the neighborhood.

Image Smoothing

- Computing the mean intensity value over the neighborhood defined by the kernel:
  - Often described as *mean filters*.
  - Large kernels, or the repeated application of a small kernel produces more pronounced smoothing.

Example of Image Smoothing



Gaussian Filters

- The Gaussian filter is a smoothing filter with a nonuniform kernel, whose coefficients are derived from a 2D Gaussian function.

- The kernel coefficients diminish in size with increasing distance from the kernels centre

Gaussian Filters

$$w(i,j) = \exp\left[\frac{-(i^2 + j^2)}{2\sigma^2}\right]$$

$$for\; i,j = -[3\sigma],\ldots,[3\sigma]$$

- [3σ] denotes the "integer part" of 3σ
- Limits of ±3 are chosen because Gaussian weights are negigible beyond them

Gaussian Filters

- If $\sigma^2 = 1$ then the normalised Gaussian kernel is:

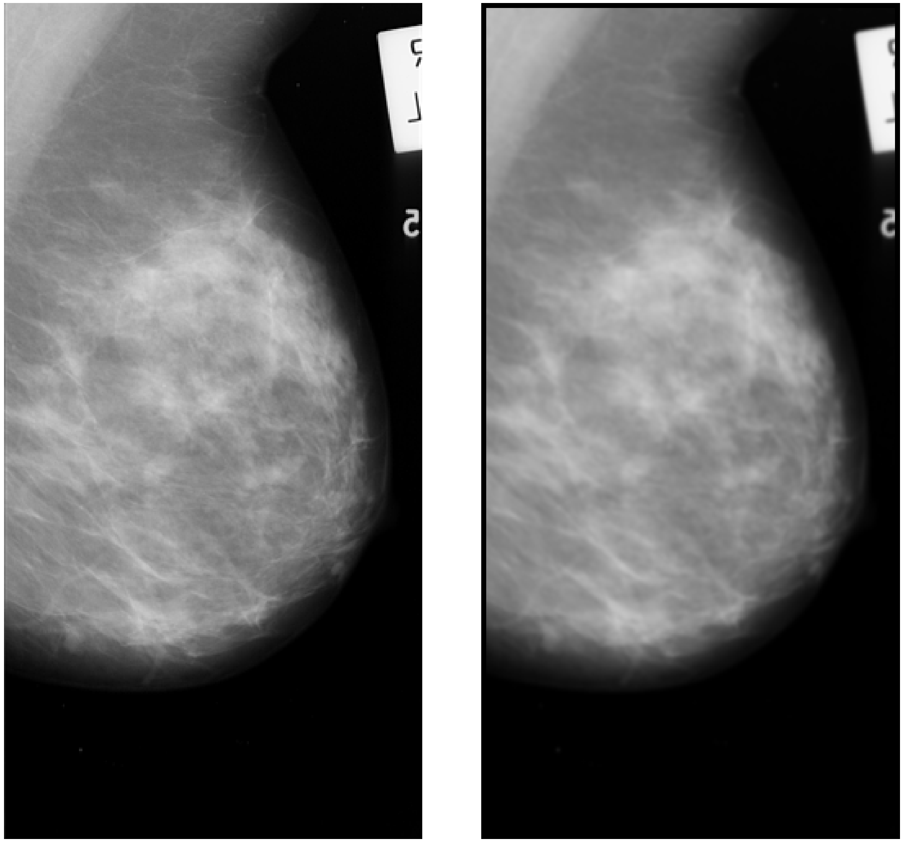
$$w(i,j) = \frac{1}{1000}\begin{bmatrix} 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 3 & 13 & 22 & 13 & 3 & 0 \\ 1 & 13 & 59 & 97 & 59 & 13 & 1 \\ 2 & 22 & 97 & 159 & 97 & 22 & 2 \\ 1 & 13 & 59 & 97 & 59 & 13 & 1 \\ 0 & 3 & 13 & 22 & 13 & 3 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \end{bmatrix}$$

Other common values for $\sigma^2 = \frac{2}{3}, 2, 6\frac{2}{3}$

Gaussian Filters

- Gaussian characteristics:
  - More weight is given to central pixels than to those in the periphery of the neighborhood
  - Large values of σ produce a wider peak → increased blurring
  - As σ increases the dimensions of the kernel also increase
  - The kernel is rotationally symmetric, so there is no directional bias in the amount of smoothing
  - The Gaussian kernel is separable.

Gaussian Filters



Median Filter

- Noise tends to spread outwards when convolution is applied.
- In a median filter pixels are replaced by the median value of the neighboring intensities in a $k \times k$ neighborhood.
- Efficient in eliminating isolated and impulse (i.e. salt-and-pepper) noise.

Median Filter

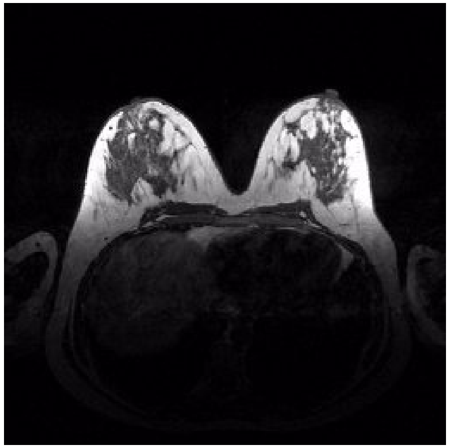
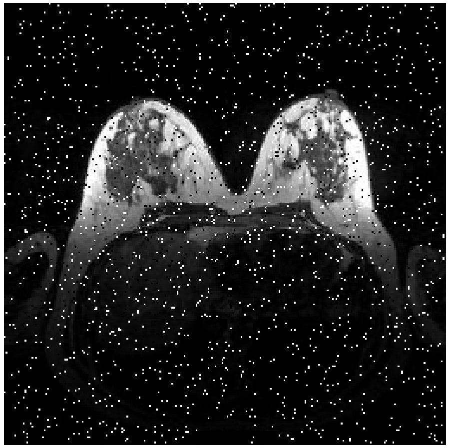
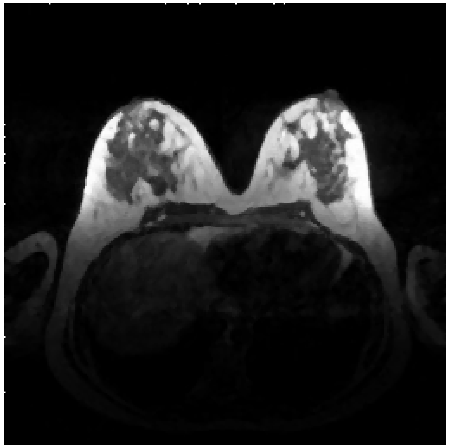
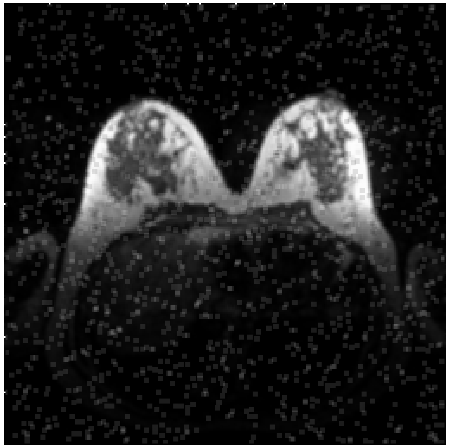





Image Sharpening

- Image sharpening, or high-pass filtering allows high spatial frequencies to remain unchanged, but suppresses low frequencies.
  - A high-pass filter has the effect of preserving sudden variations in intensity, such as those that occur at the boundaries of objects, but suppresses more gradual variations.
  - Makes noise more prominent (noise has a strong high-frequency component).

Image Sharpening

- A HPF convolution kernel contains a mixture of positive and negative values.
- An omnidirectional high pass filter should have positive coefficients near its centre and negative coefficients in the periphery of the kernel.

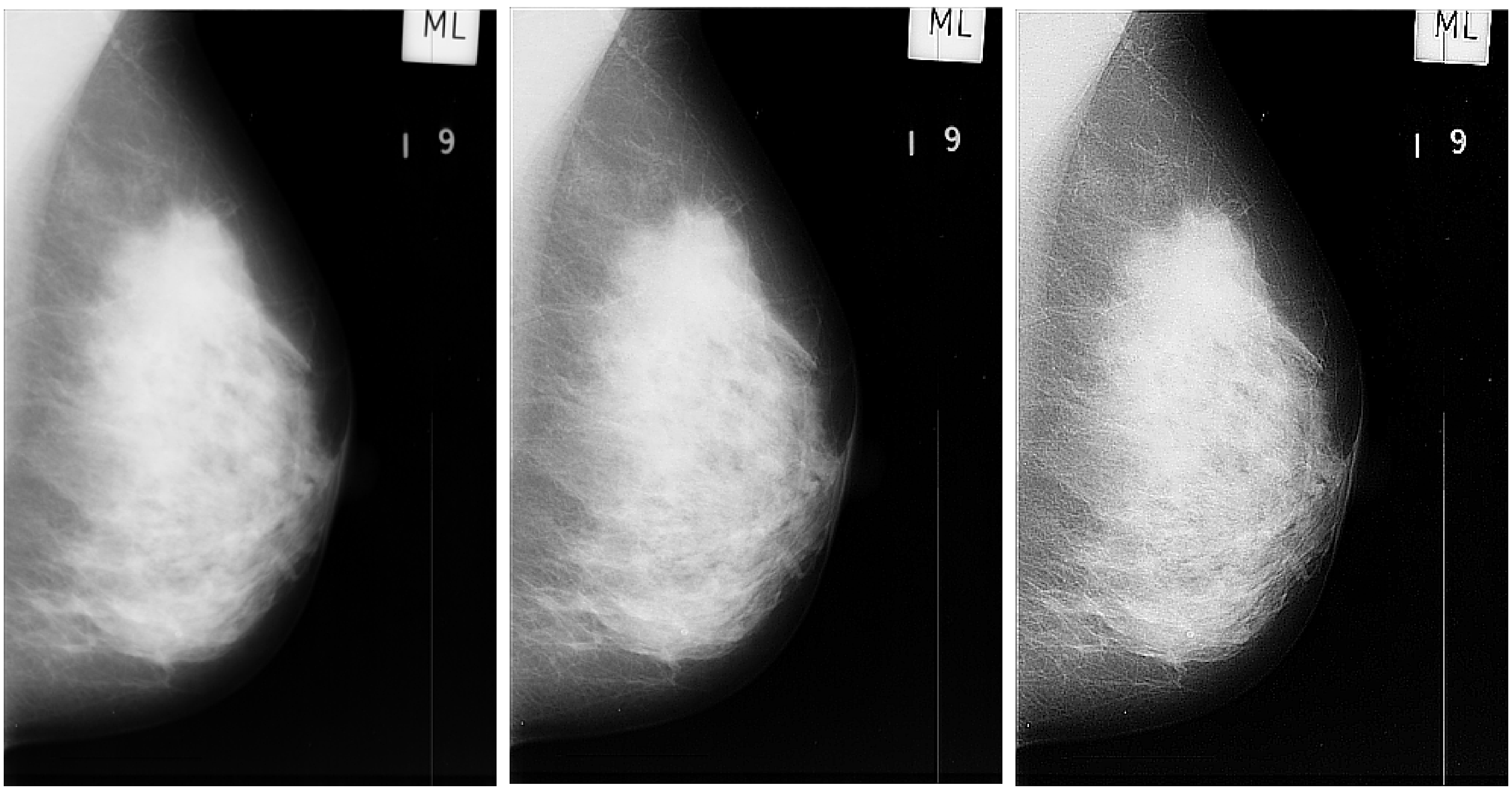
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

  – The sum of the coefficients in this kernel is zero.

Image Sharpening

- When the kernel is over an area of constant or slow-changing intensity values the result of the convolution is zero (or small).
- When the intensity values vary rapidly within a neighborhood, the result of the convolution is a large number.
- The result can be positive or negative:
  – Map the pixel values onto a 0-255 range. A filter response of 0 maps onto the middle of the range.

Example of Image Sharpening



Unsharp Masking

- Subtracting from an image a "blurred" version of that image (thereby removing the low spatial frequencies) is known as *unsharp masking*. $\hat{f}(x,y) = f(x,y) - S(f(x,y))$

High-Boost Filter

- Compute a weighted sum of the original image and the output from a high-pass filter.
  - The result is an image in which high spatial frequencies are emphasized relative to lower frequencies.
  - The high-boost filter is used to sharpen an image.
  - Can be performed in a single convolution operation

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & c & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad (c > 8)$$