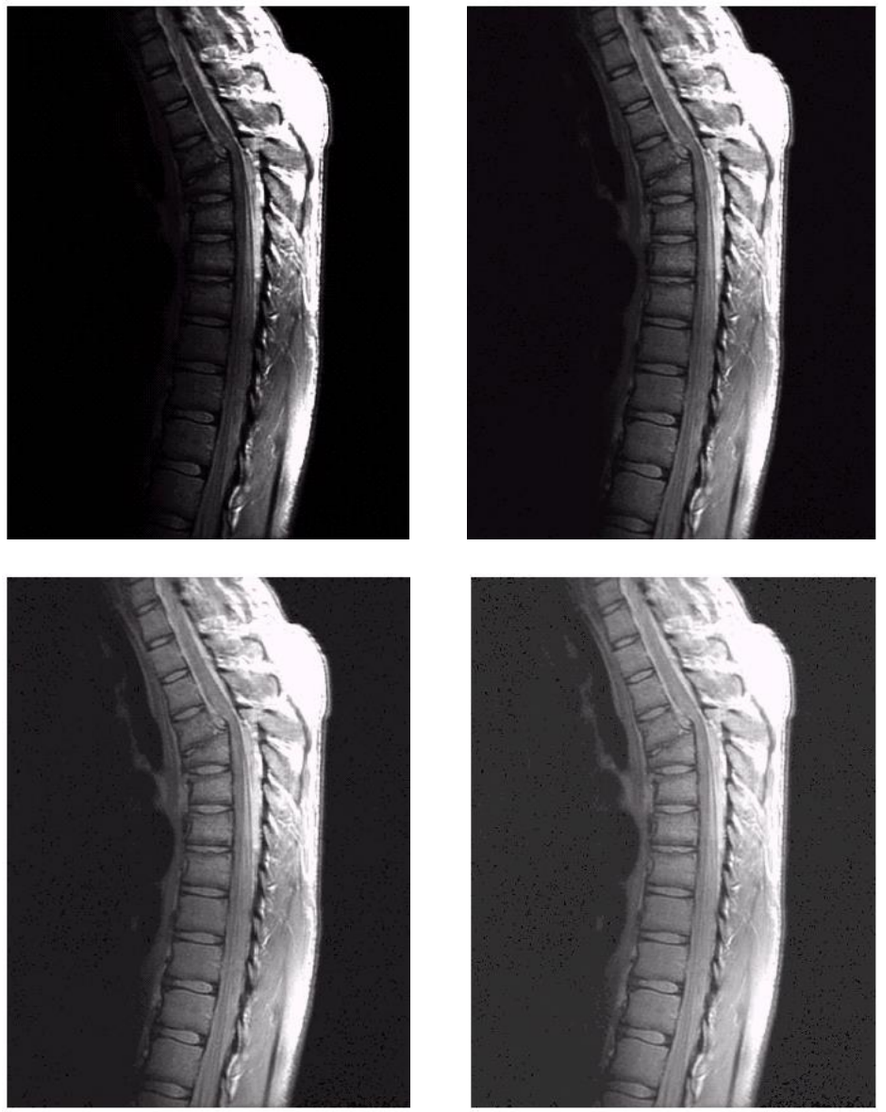
Image Processing and Pattern Recognition (IPPR)

Lecture 4

Image Enhancement in Spatial Domain Cont…



Basanta Joshi, PhD

Asst. Prof., Depart of Electronics and Computer Engineering

Program Coordinator, MSc in Information and Communication Engineering

Member, Laboratory for ICT Research and Development (LICT)

Member, Research Management Cell (RMC)

Institute of Engineering

basanta@ioe.edu.np

http://www.basantajoshi.com.np

https://scholar.google.com/citations?user=iocLiGcAAAAJ
https://www.researchgate.net/profile/Basanta_Joshi2

What Is Image Enhancement?

Image enhancement is the process of making images more useful

The reasons for doing this include:

- Highlighting interesting detail in images
- Removing noise from images
- Making images more visually appealing

Spatial & Frequency Domains

There are two broad categories of image enhancement techniques

- Spatial domain techniques
  - Direct manipulation of image pixels
- Frequency domain techniques
  - Manipulation of Fourier transform or wavelet transform of an image

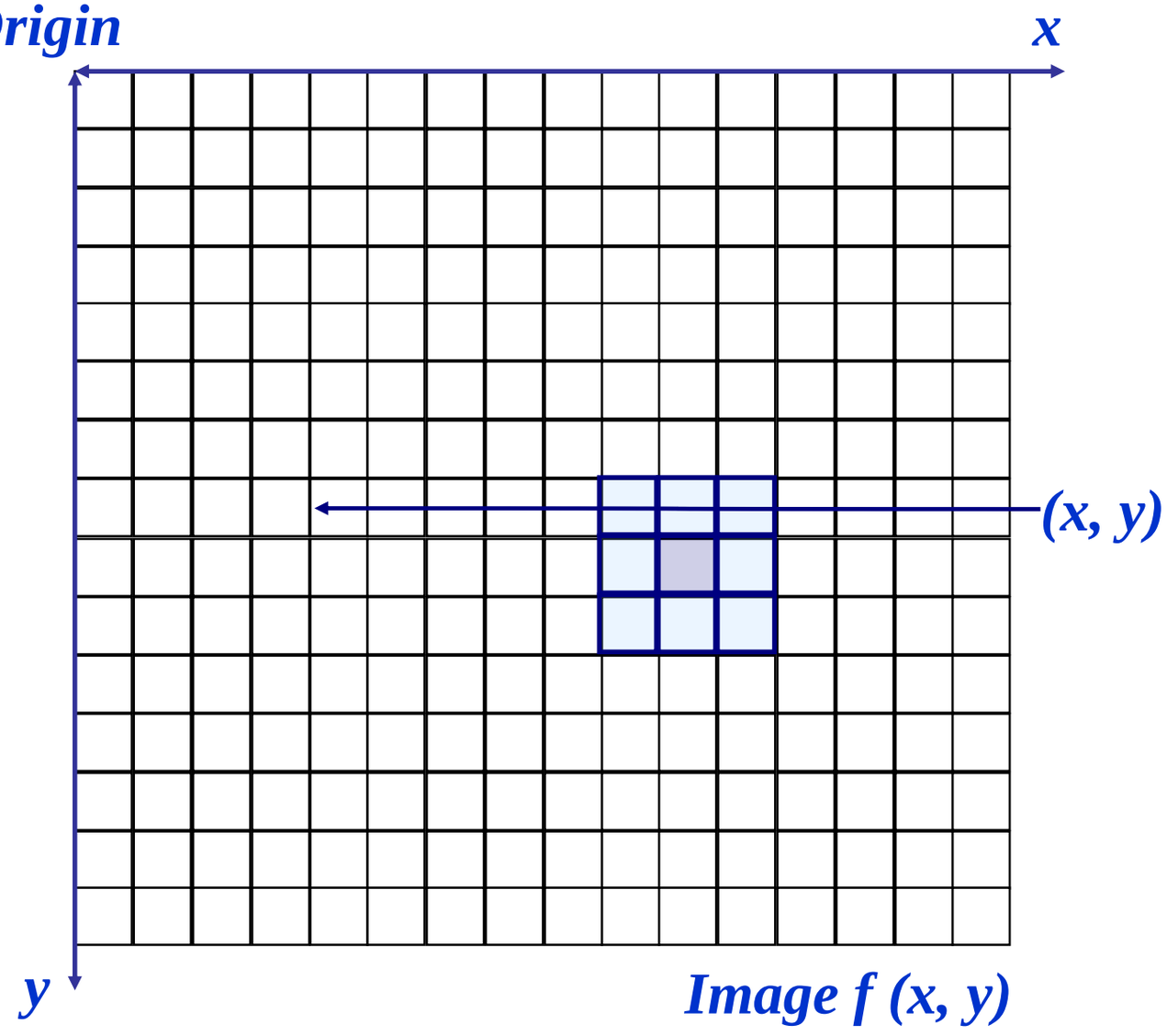
For the moment we will concentrate on techniques that operate in the spatial domain

Basic Spatial Domain Image Enhancement

Most spatial domain enhancement operations can be reduced to the form

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

where $f(x, y)$ is the input image, $g(x, y)$ is the processed image and T is some operator defined over some neighbourhood of (x, y)

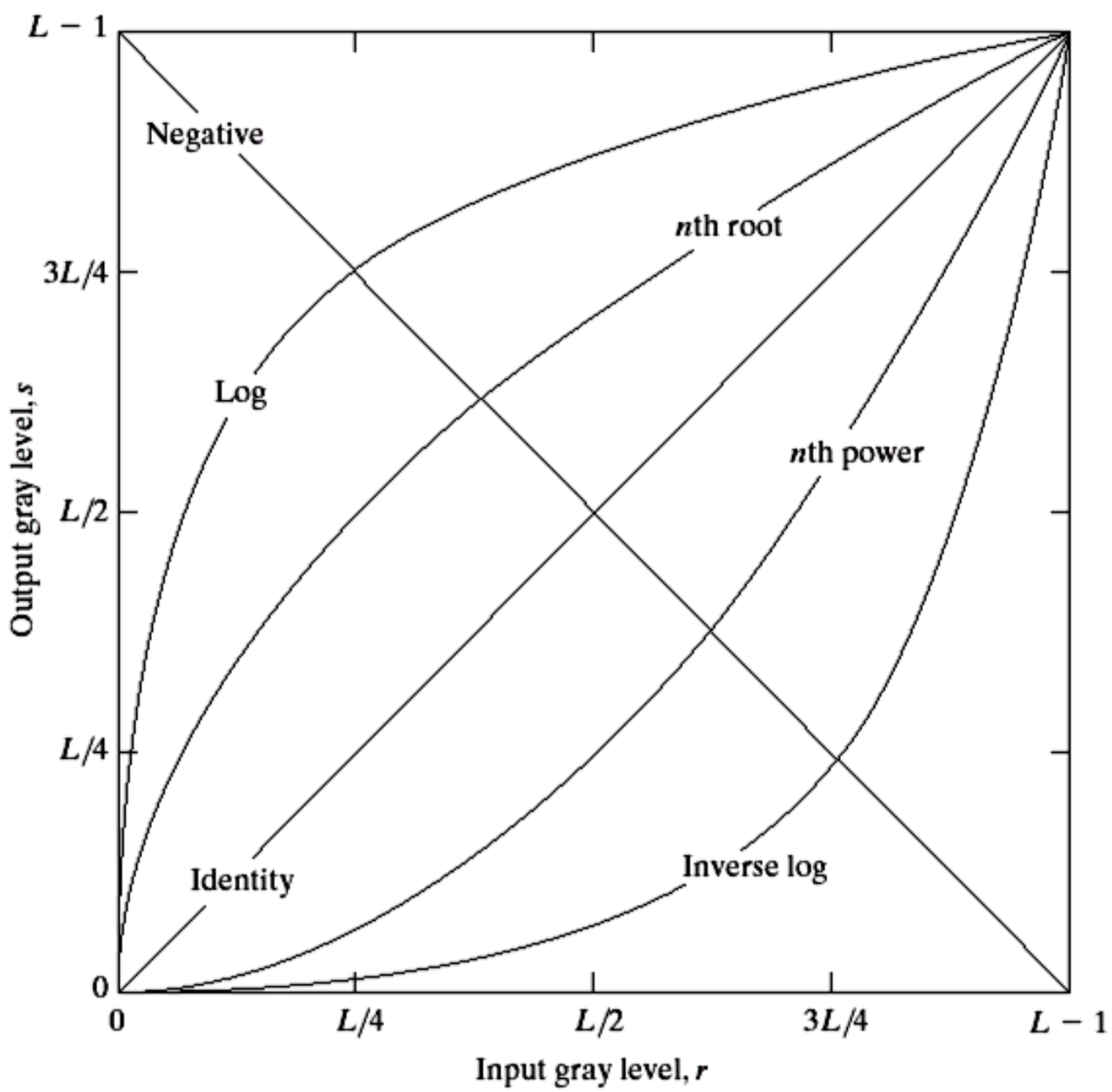


Basic Grey Level Transformations

There are many different kinds of grey level transformations

Three of the most common are shown here

- Linear
  - Negative/Identity
- Logarithmic
  - Log/Inverse log
- Power law
  - n^{th} power/n^{th} root



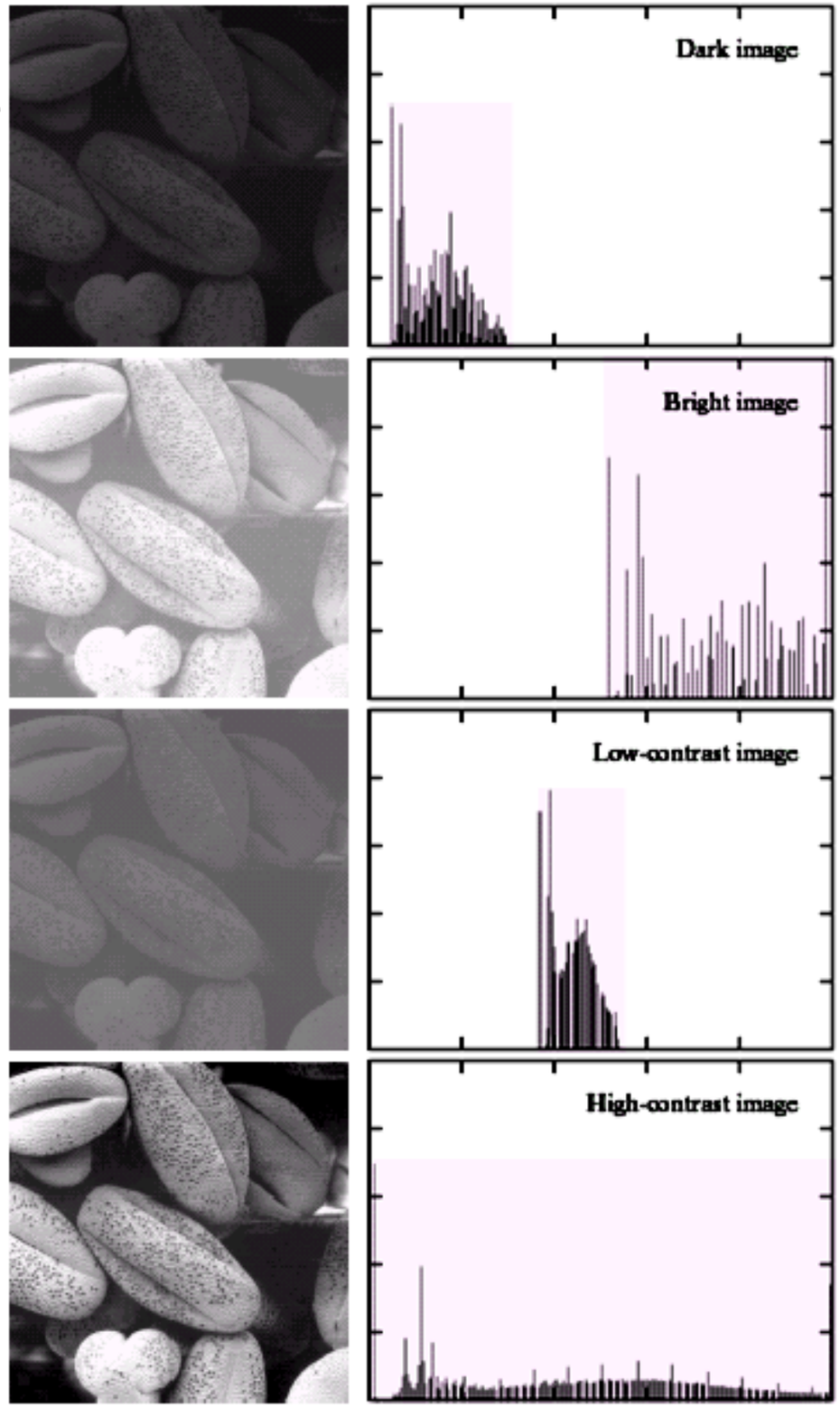
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Histogram

The histogram of an image distribution of grey levels in

Notice the relationships between the images and their histograms

Note that the high contrast image has the most evenly spaced histogram



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Spatial Filtering

• Some neighborhood operations work with the values of the image pixels in the neighborhood and the corresponding values of a sub-image that has the same dimensions as the neighborhood.

• The sub-image is called a filter, mask, kernel, template, or, window, with the first three terms being the most prevalent terminology.

• The values in a filter sub-image are referred to as coefficients, rather than pixels.

• Use of spatial masks for filtering is called spatial filtering

– May be linear or nonlinear

a	b	c
d	e	f
g	h	i

Image Pixels with neighborhood

r	s	t
u	v	w
x	y	z

Sub-image

Spatial Filtering

- **Linear filters**

  – Lowpass: attenuate (or eliminate) high frequency components such as characterized by edges and sharp details in an image

    - Net effect is image blurring

  – Highpass: attenuate (or eliminate) low frequency components such as slowly varying characteristics

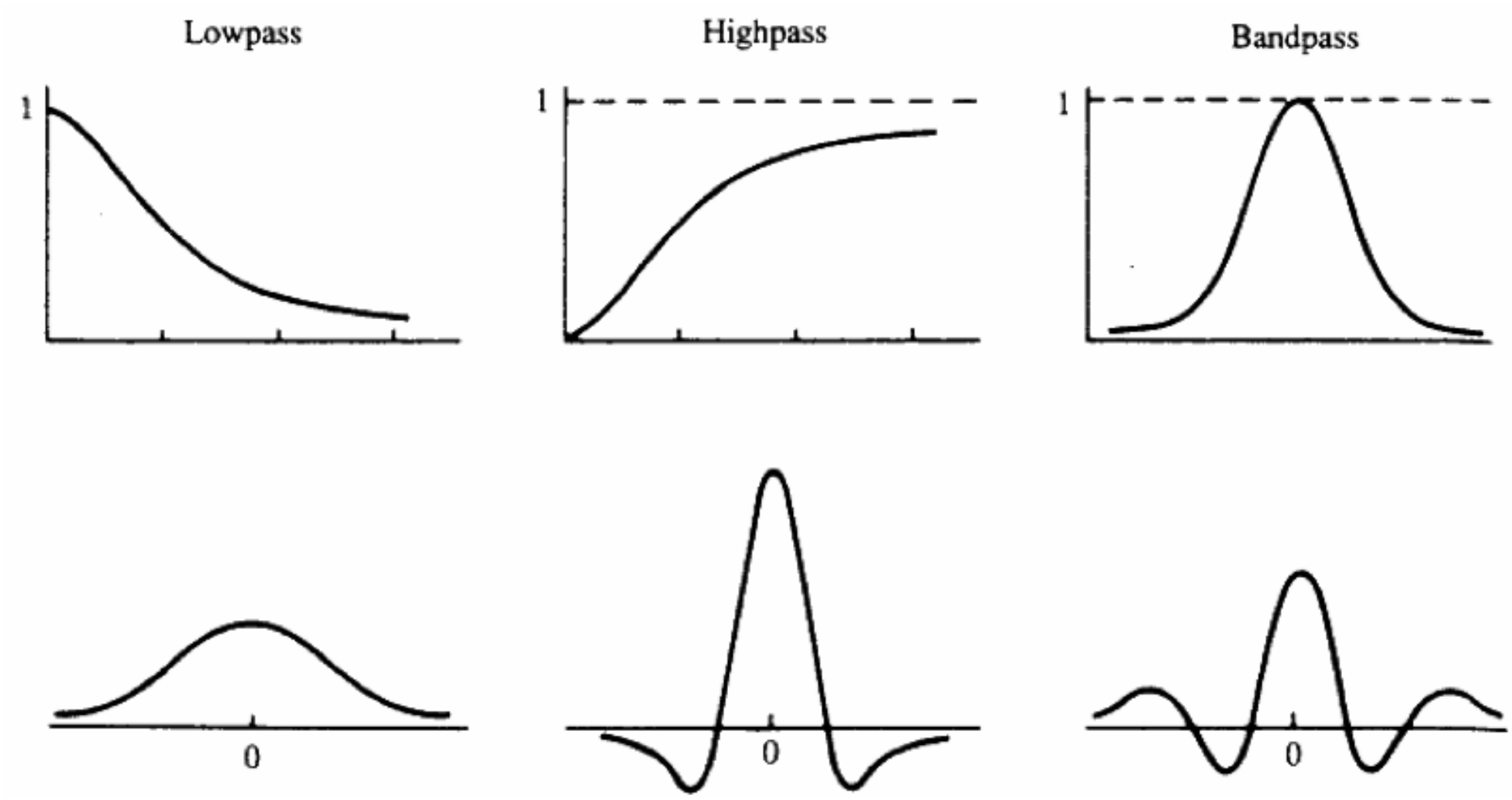
    - Net effect is a sharpening of edges and other details

  – Bandpass: attenuate (or eliminate) a given frequency range

    - Used primarily for image restoration(are of little interest for image enhancement)

Spatial Filtering

- Filters in the frequency domain and corresponding spatial filters
- Basic approach is to sum products between mask coefficients and pixel values
  - $R=w_1z_1 +w_2z_2 +....+w_9z_9$



Order-Statistic nonlinear spatial filters

- Nonlinear spatial filters also operate on neighborhoods
- Their operation is based directly on pixel values in the neighborhood under consideration

  – They do not explicitly use coefficient values as in the linear spatial filters

- Example nonlinear spatial filters

  – Median filter: Computes the median gray-level value of the neighborhood. Used for noise reduction.

  – Max filter: Used to find the brightest points in an image

$$R=\max\{z_k \mid k=1,2,...,9\}$$

  – Min filter: Used to find the dimmest points in an image

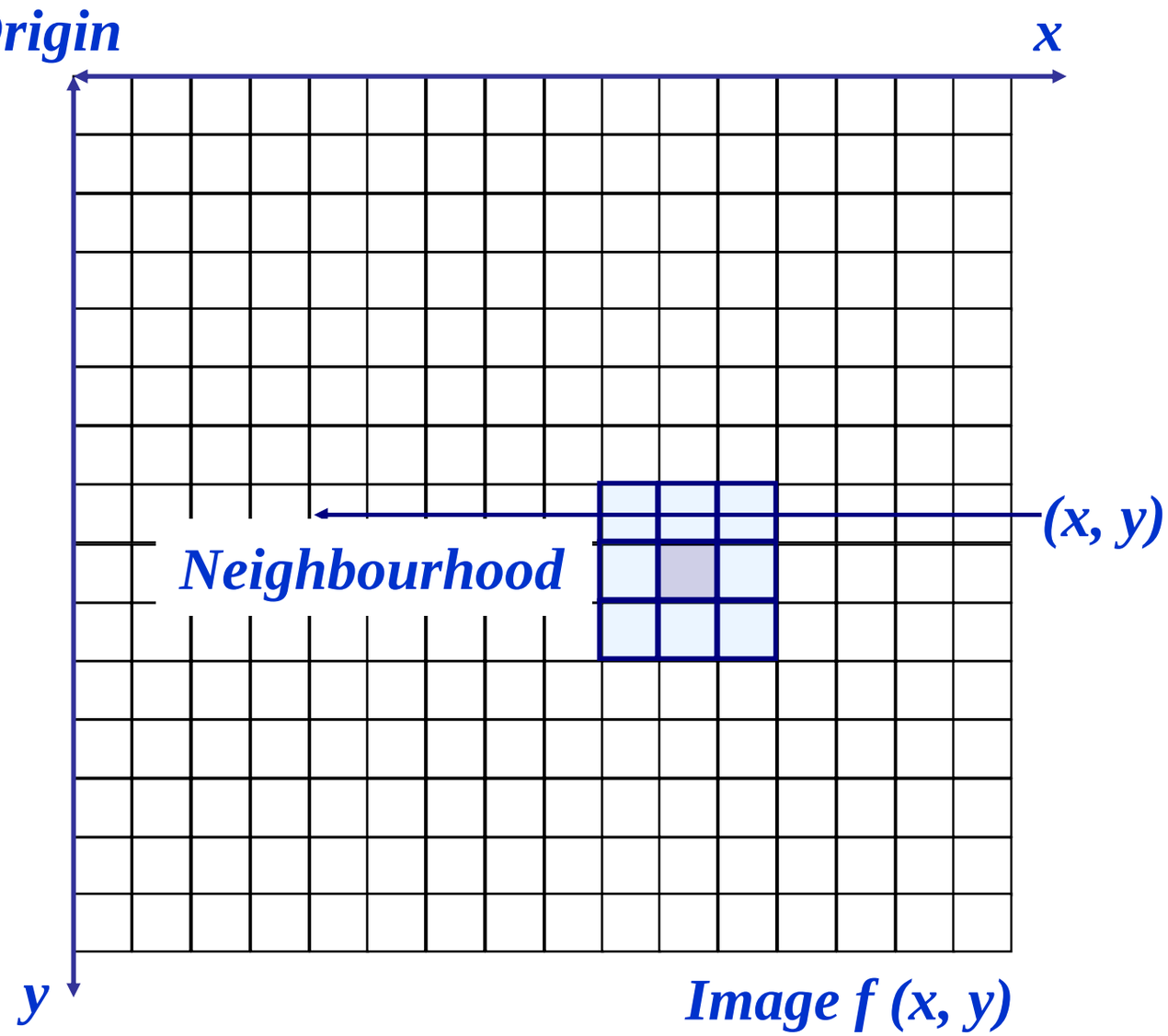
$$R=\min\{z_k \mid k=1,2,...,9\}$$

Neighbourhood Operations

Neighbourhood operations simply operate on a larger neighbourhood of pixels than point operations

Neighbourhoods are mostly a rectangle around a central pixel

Any size rectangle and any shape filter are possible



Simple Neighbourhood Operations

Some simple neighbourhood operations include:

Min: Set the pixel value to the minimum in the neighbourhood

Max: Set the pixel value to the maximum in the neighbourhood

Median: The median value of a set of numbers is the midpoint value in that set (e.g. from the set [1, 7, 15, 18, 24] 15 is the median). Sometimes the median works better than the average

Simple Neighbourhood Operations Example

Original Image

x

123	127	128	119	115	130
140	145	148	153	167	172
133	154	183	192	194	191
194	199	207	210	198	195
164	170	175	162	173	151

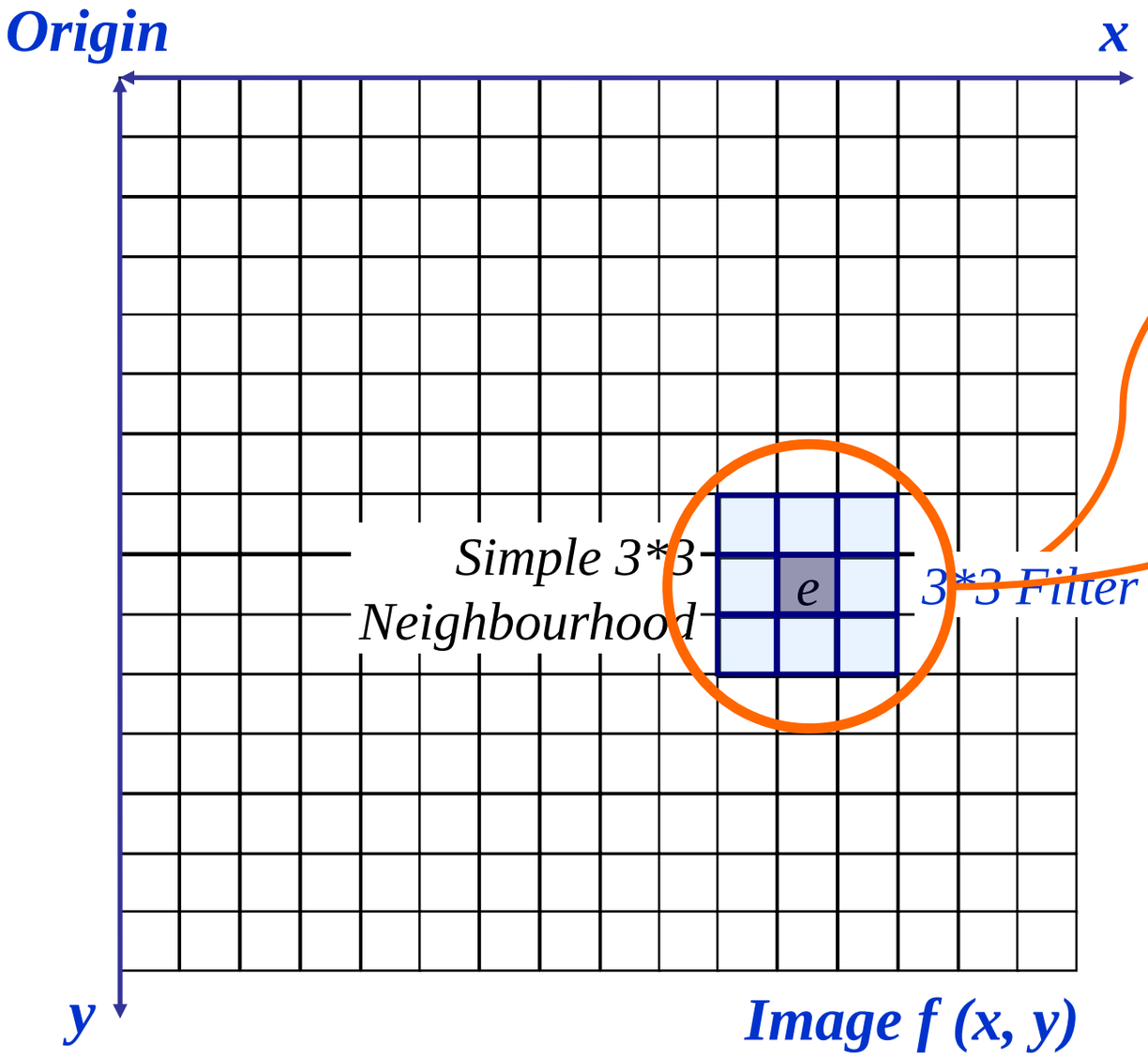
y

Enhanced Image

x

y

The Spatial Filtering Process



a	*b*	*c*
d	*e*	*f*
g	*h*	*i*

Original Image Pixels

*

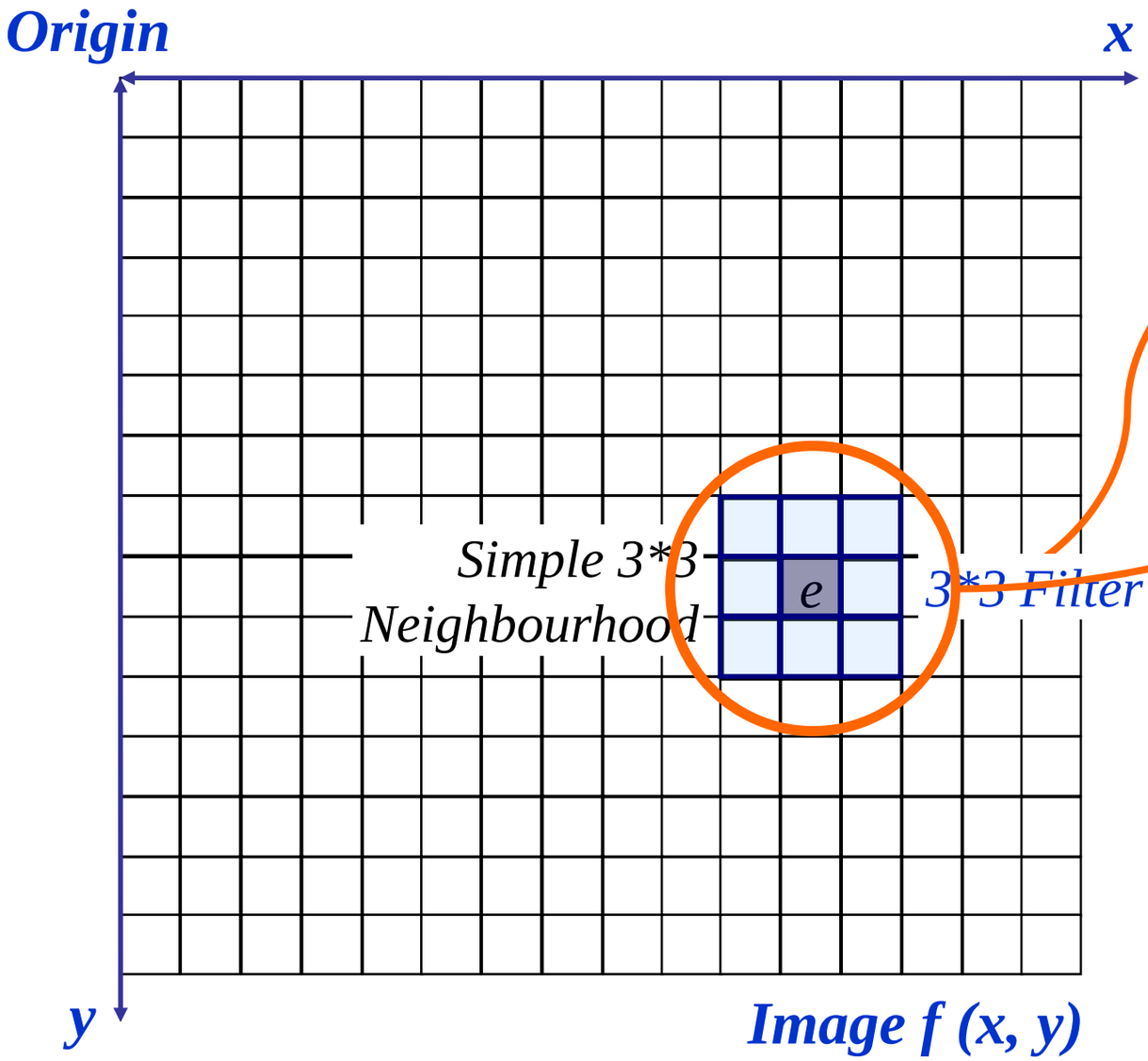
r	*s*	*t*
u	*v*	*w*
x	*y*	*z*

Filter

$$e_{processed} = v*e + r*a + s*b + t*c + u*d + w*f + x*g + y*h + z*i$$

The above is repeated for every pixel in the original image to generate the filtered image

The Spatial Filtering Process



z_1	z_2	z_3
z_4	z_4	z_5
z_6	z_7	z_8

Original Image Pixels

$*$

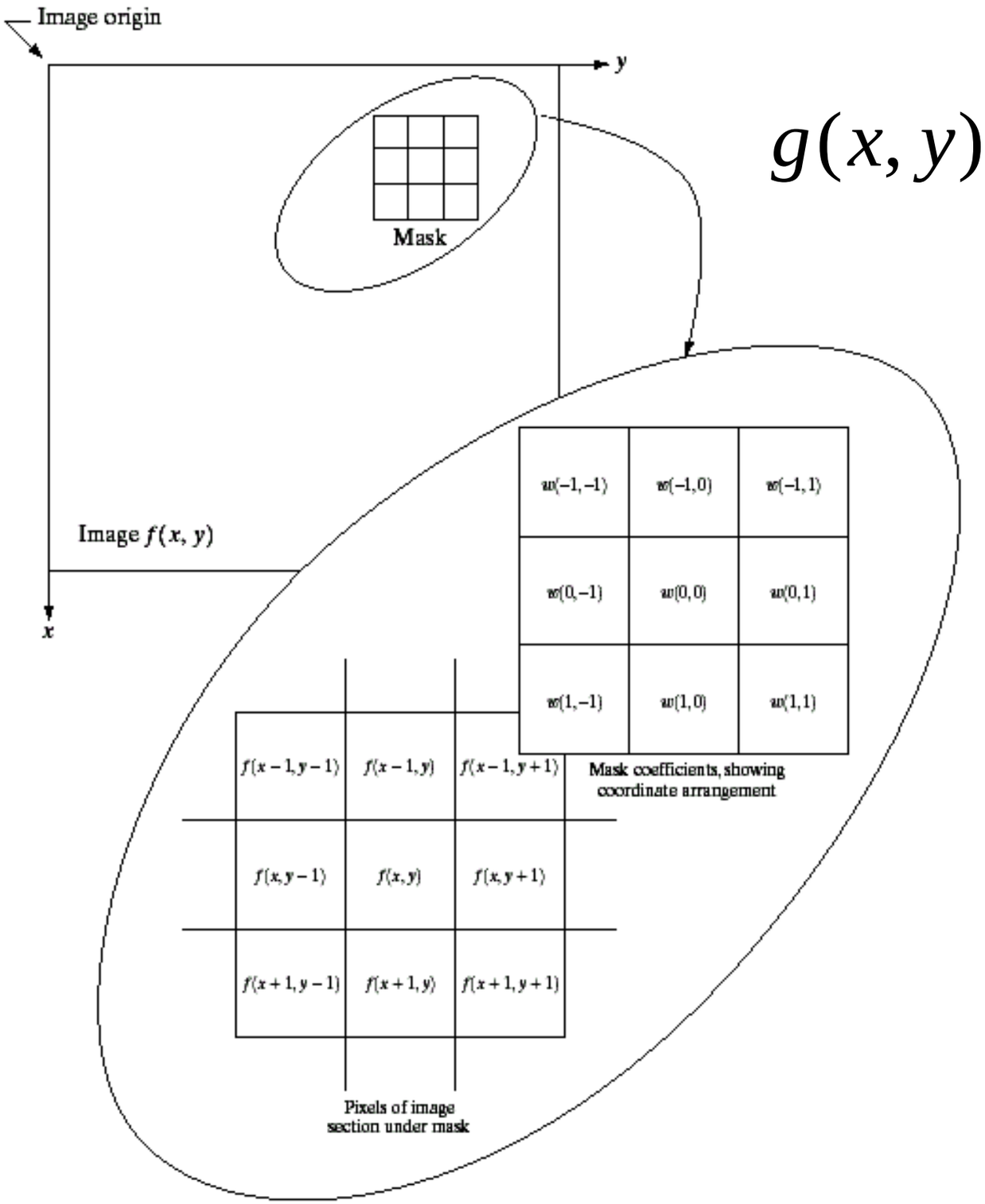
w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Filter

$$e_{processed} = z_1*w_1 + z_2*w_{2+} z_3*w_{3+} z_4*w_4 + z_5*w_5 + z_6*w_6 + z_7*w_7 + z_8*w_8 + z_9*w_9$$

The above is repeated for every pixel in the original image to generate the filtered image

Spatial Filtering: Equation Form



$$g(x,y) = \sum_{s=-a}^{a}\sum_{t=-b}^{b} w(s,t)\, f(x+s, y+t)$$

Filtering can be given in equation form as shown above

Notations are based on the image shown to the left

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Smoothing Spatial Filters

• The shape of the impulse response needed to implement a lowpass (smoothing) filter indicates the filter should have all positive coefficients

• For a 3x3 mask, the simplest arrangement is to have all the coefficient values equal to one (neighborhood averaging)

– The response would be the sum of all gray levels for the nine pixels in the mask

– This could cause the value of R to be out of the valid gray-level range

• The solution is to scale the result by dividing by 9

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Smoothing Spatial Filters

One of the simplest spatial filtering operations we can perform is a smoothing operation

Simply average all of the pixels in a neighbourhood around a central value

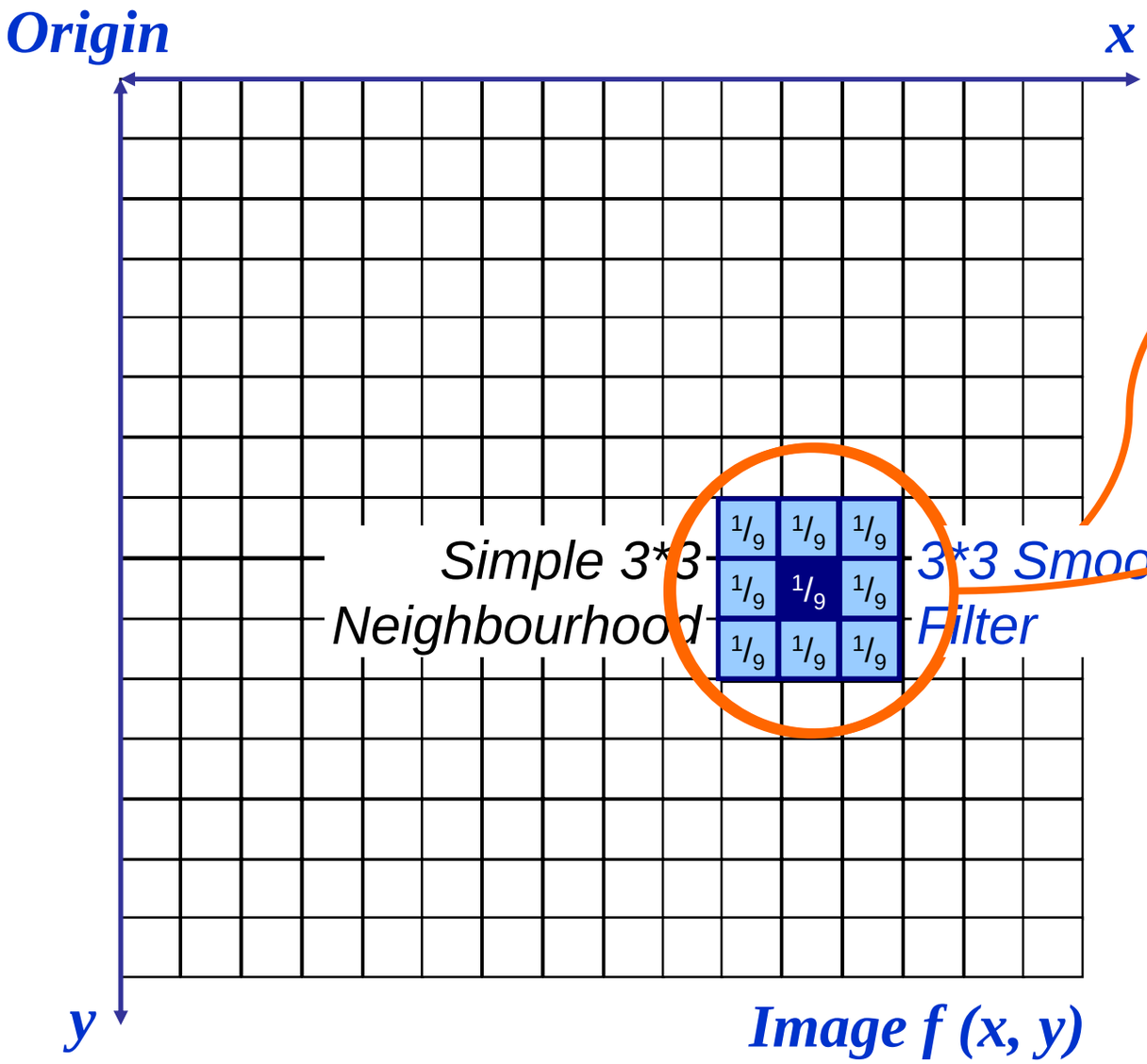
Especially useful in removing noise from images

Also useful for highlighting gross detail

$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$
$1/9$	$1/9$	$1/9$

Simple averaging filter

Smoothing Spatial Filtering



104	100	108
99	106	98
95	90	85

Original Image Pixels

*

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Filter

$$e = \frac{1}{9}*106 + \frac{1}{9}*104 + \frac{1}{9}*100 + \frac{1}{9}*108 + \frac{1}{9}*99 + \frac{1}{9}*98 + \frac{1}{9}*95 + \frac{1}{9}*90 + \frac{1}{9}*85$$

$$= 98.3333$$

The above is repeated for every pixel in the original image to generate the smoothed image

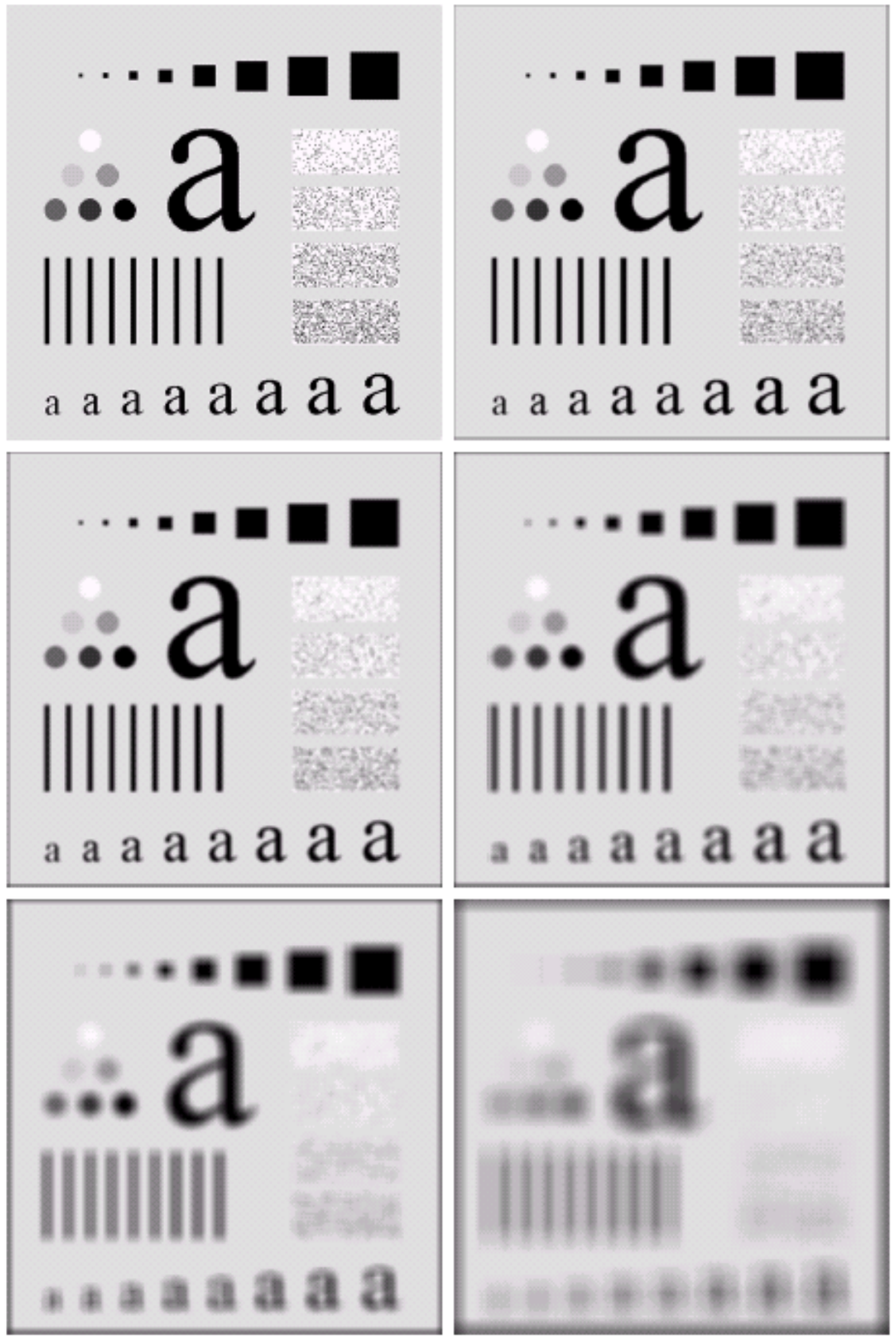
Image Smoothing Example

The image at the top left is an original image of size 500*500 pixels

The subsequent images show the image after filtering with an averaging filter of increasing sizes

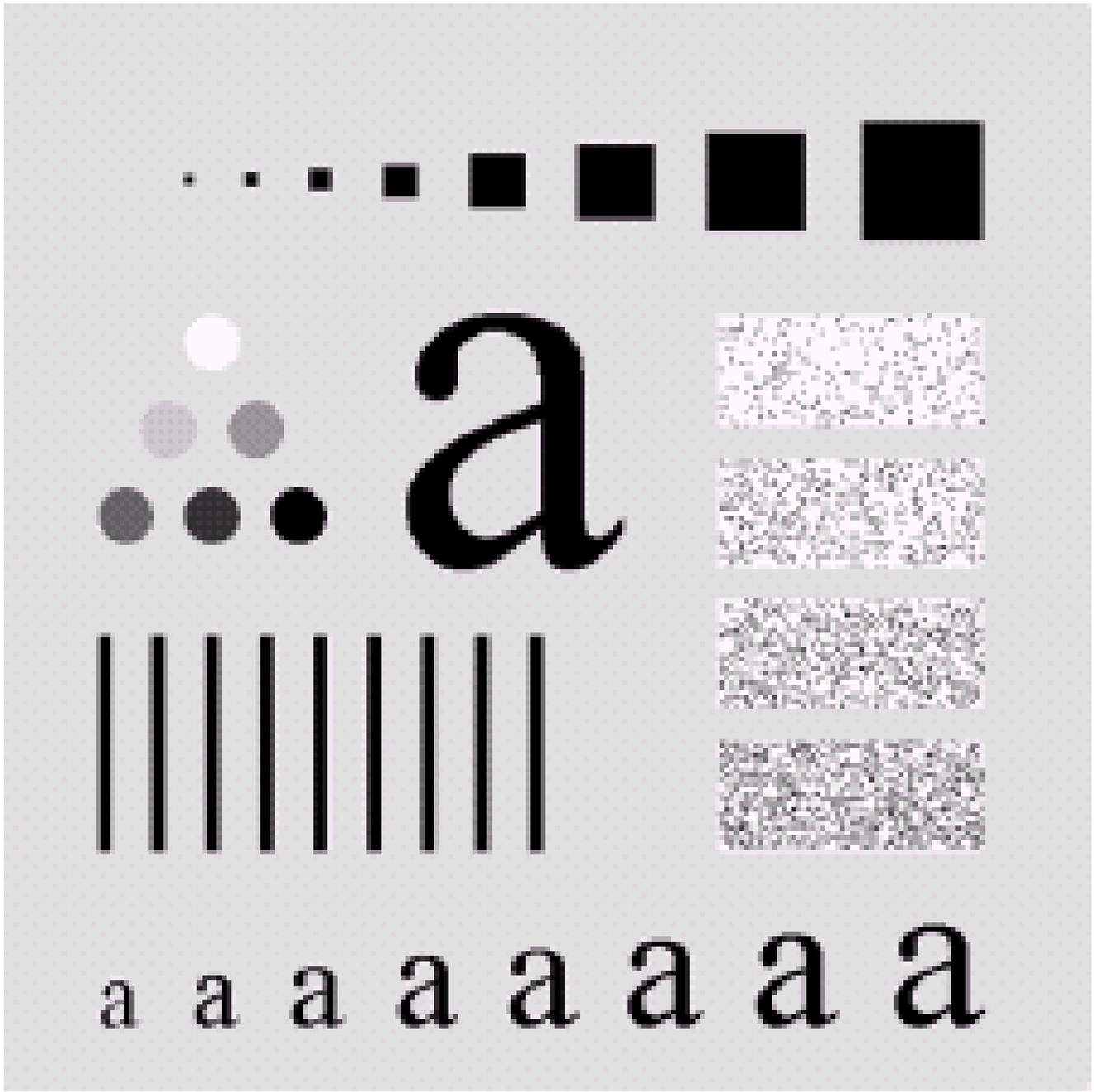
3, 5, 9, 15 and 35

Notice how detail begins to disappear



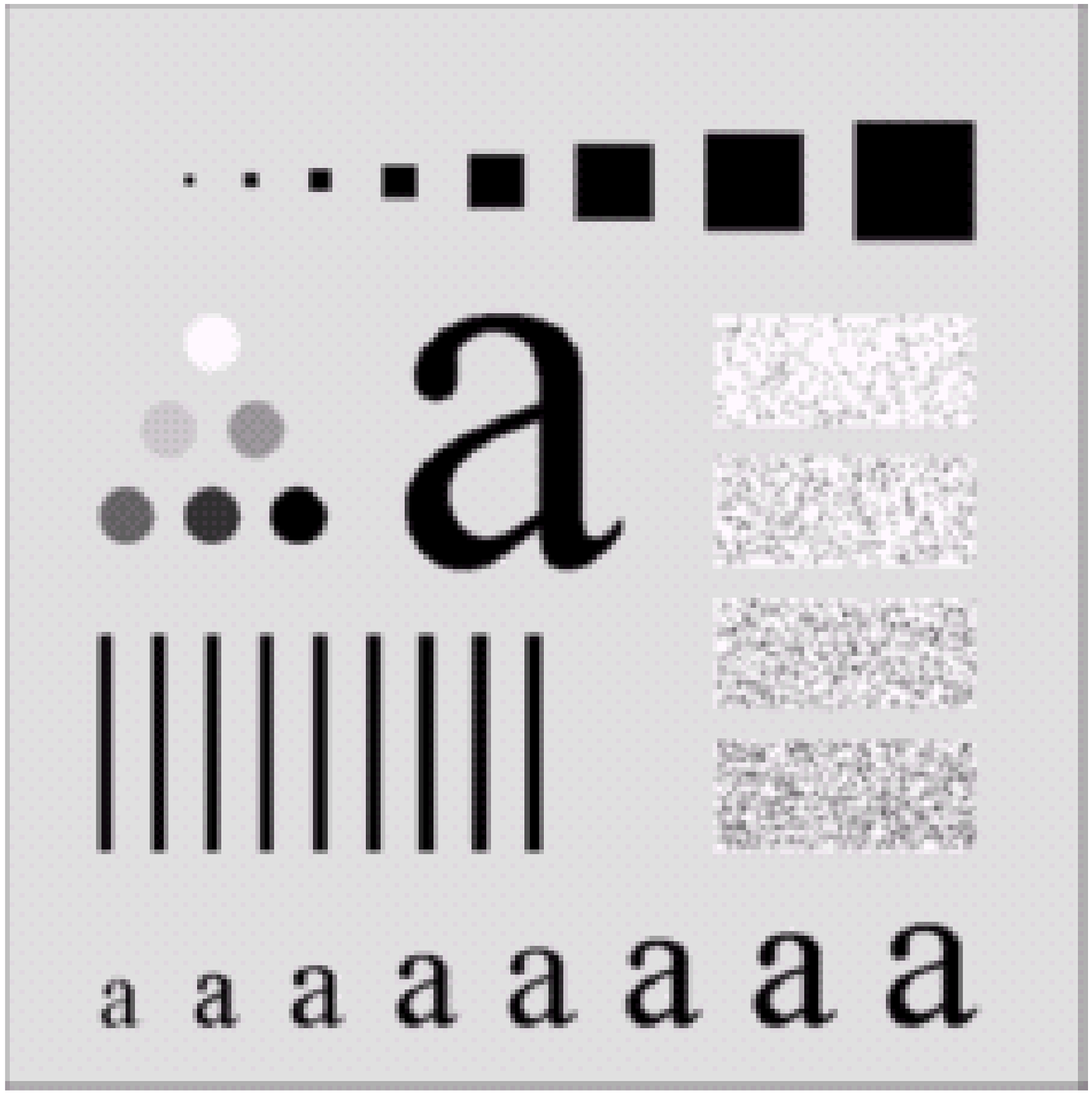
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image Smoothing Example



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image Smoothing Example

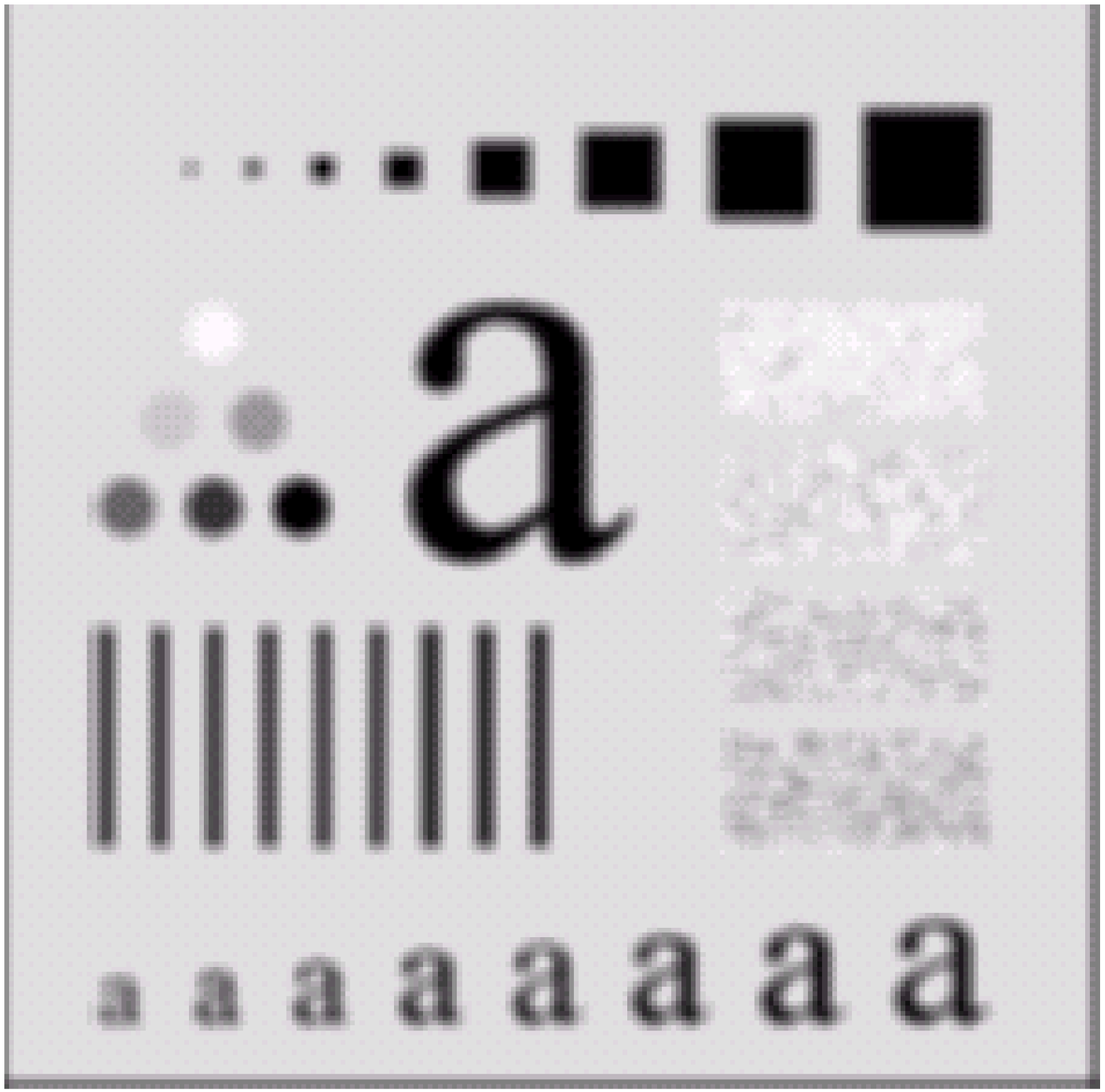


Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image Smoothing Example

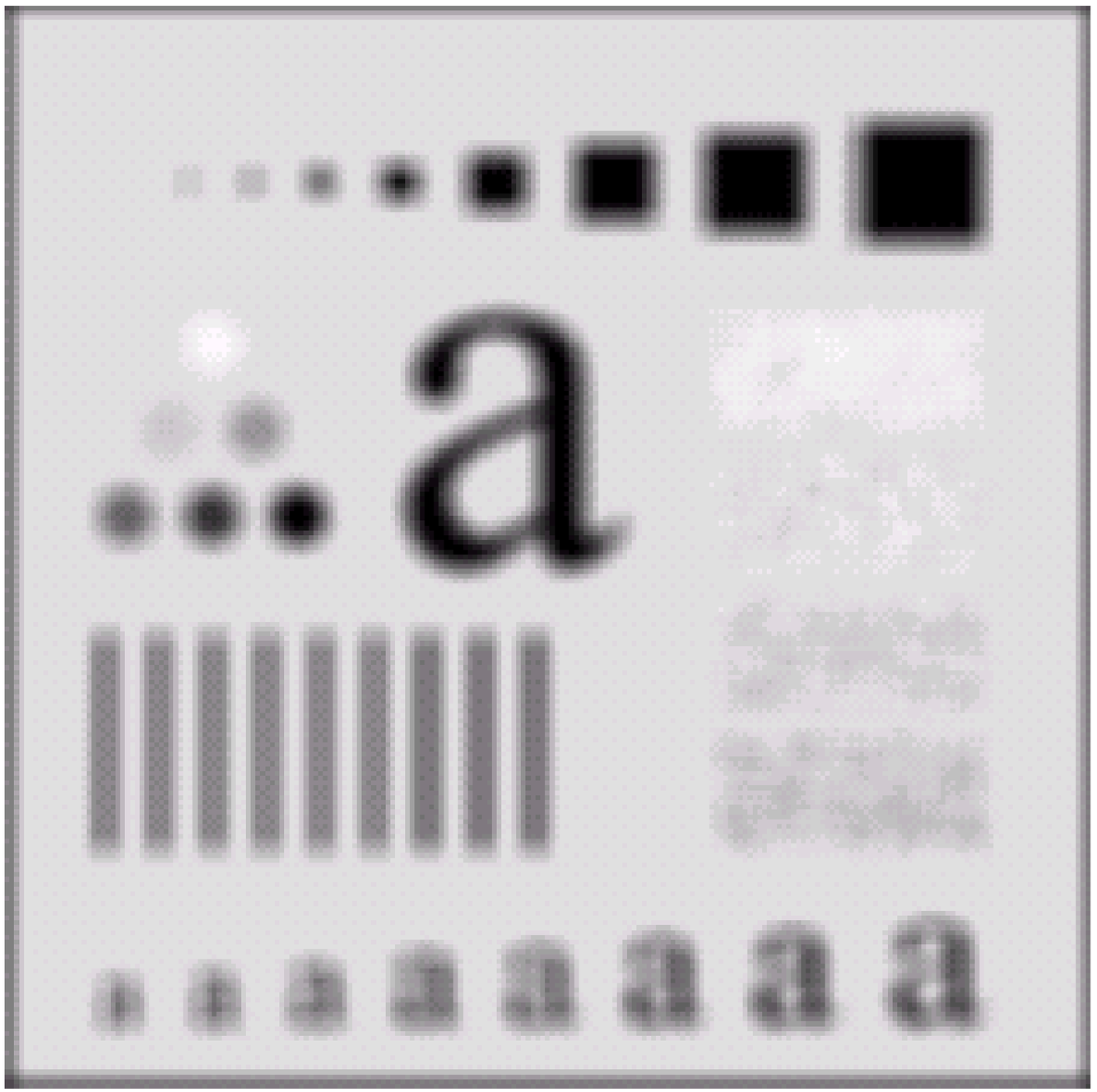
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image Smoothing Example



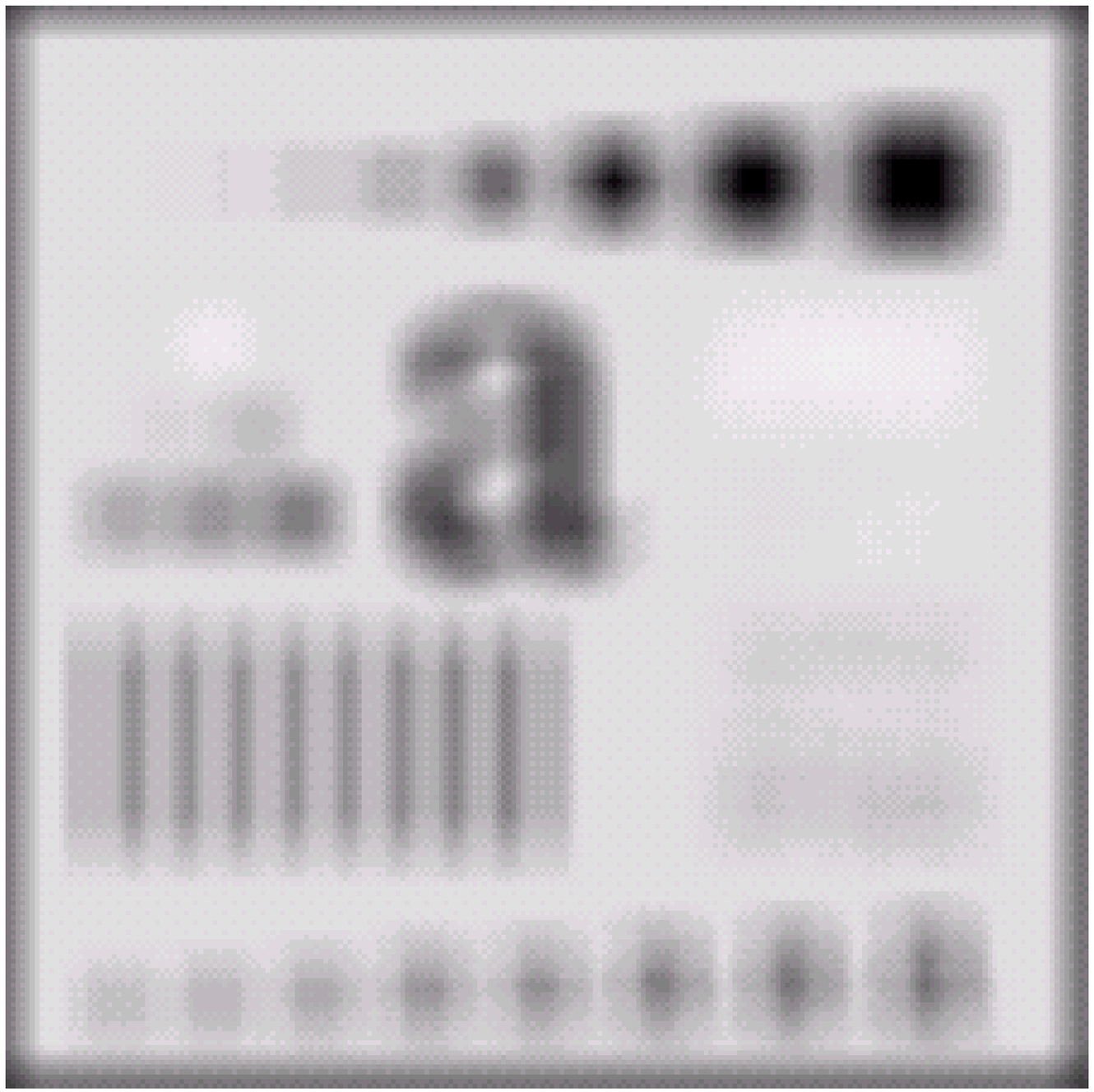
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image Smoothing Example



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image Smoothing Example

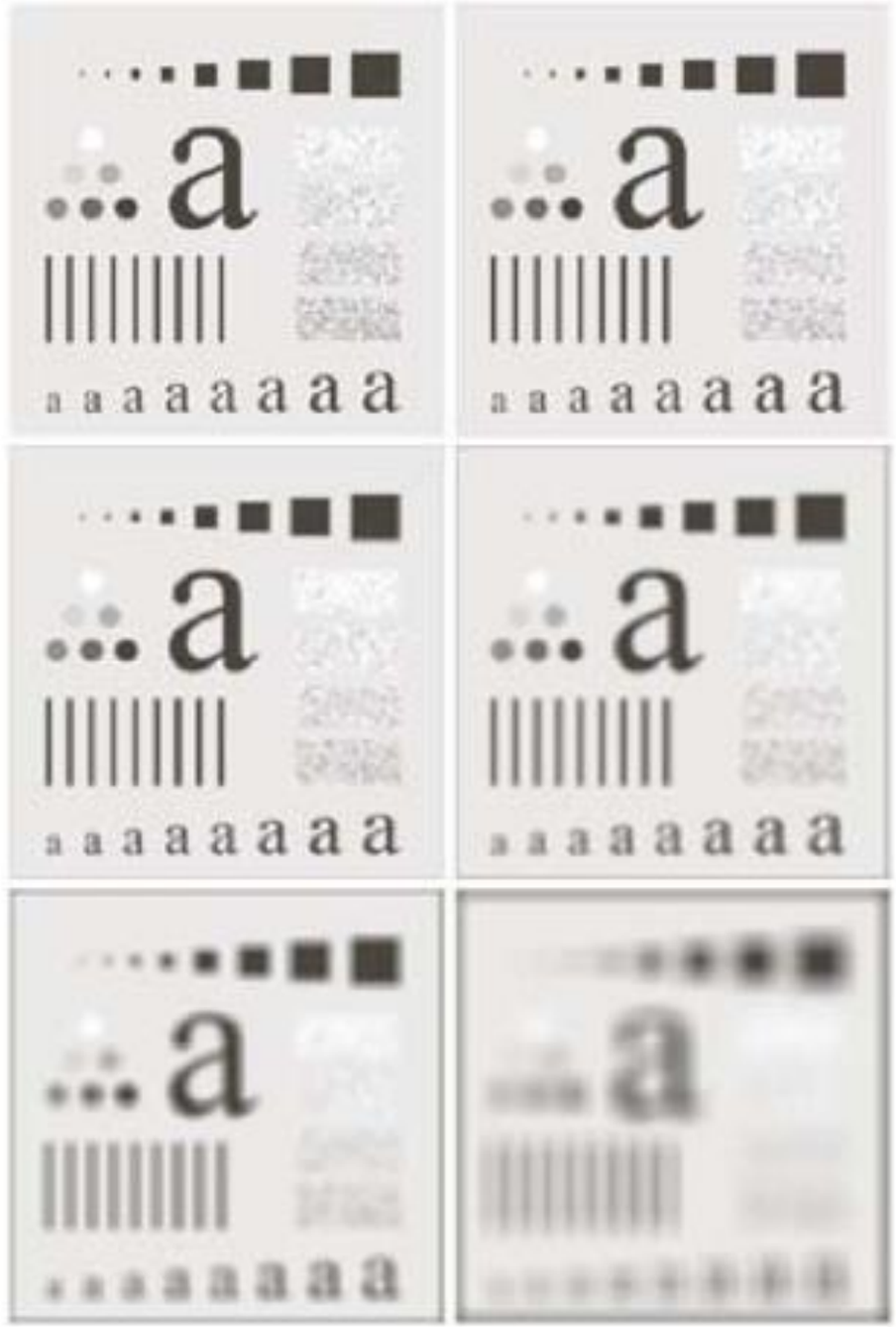


Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image Smoothing Example

FIGURE 3.33 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $m = 3, 5, 9, 15$, and 35, respectively. The black squares at the top are of sizes 3, 5, 9, 15, 25, 35, 45, and 55 pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their intensity levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.

a b
c d
e f



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Weighted Smoothing Filters

More effective smoothing filters can be generated by allowing different pixels in the neighbourhood different weights in the averaging function

Pixels closer to the central pixel are more important

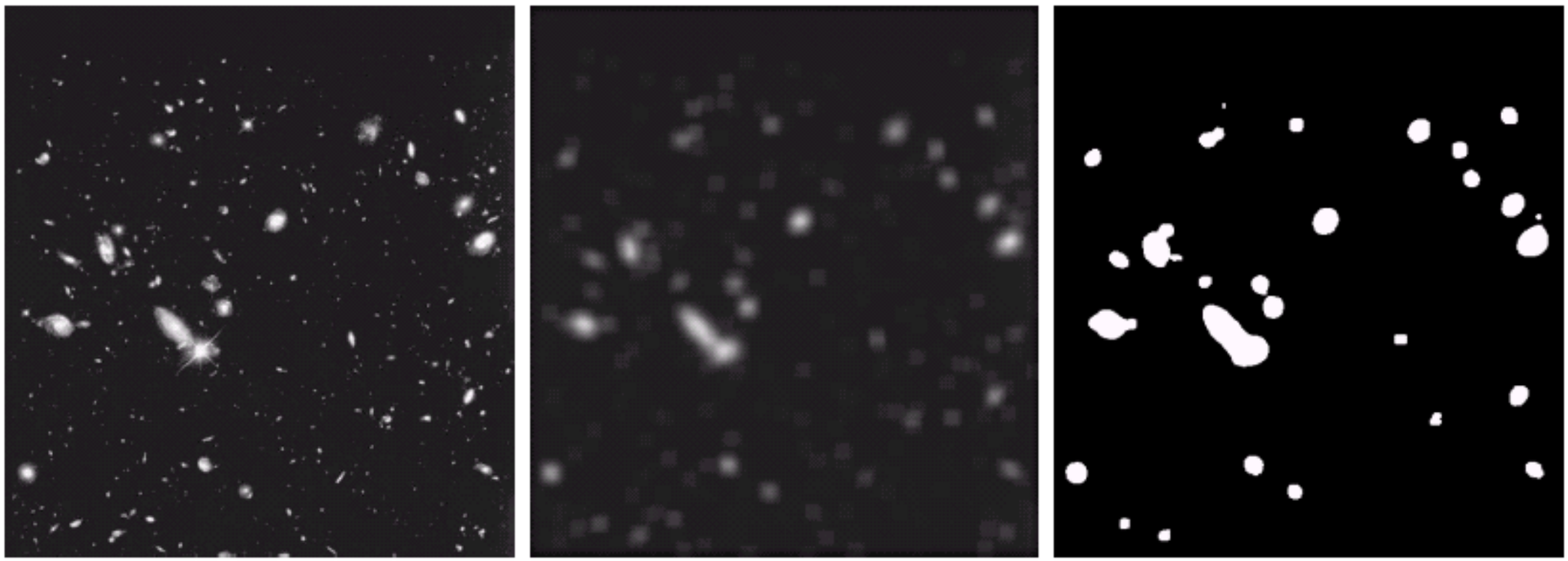
Often referred to as a *weighted averaging*

$1/16$	$2/16$	$1/16$
$2/16$	$4/16$	$2/16$
$1/16$	$2/16$	$1/16$

Weighted averaging filter

Another Smoothing Example

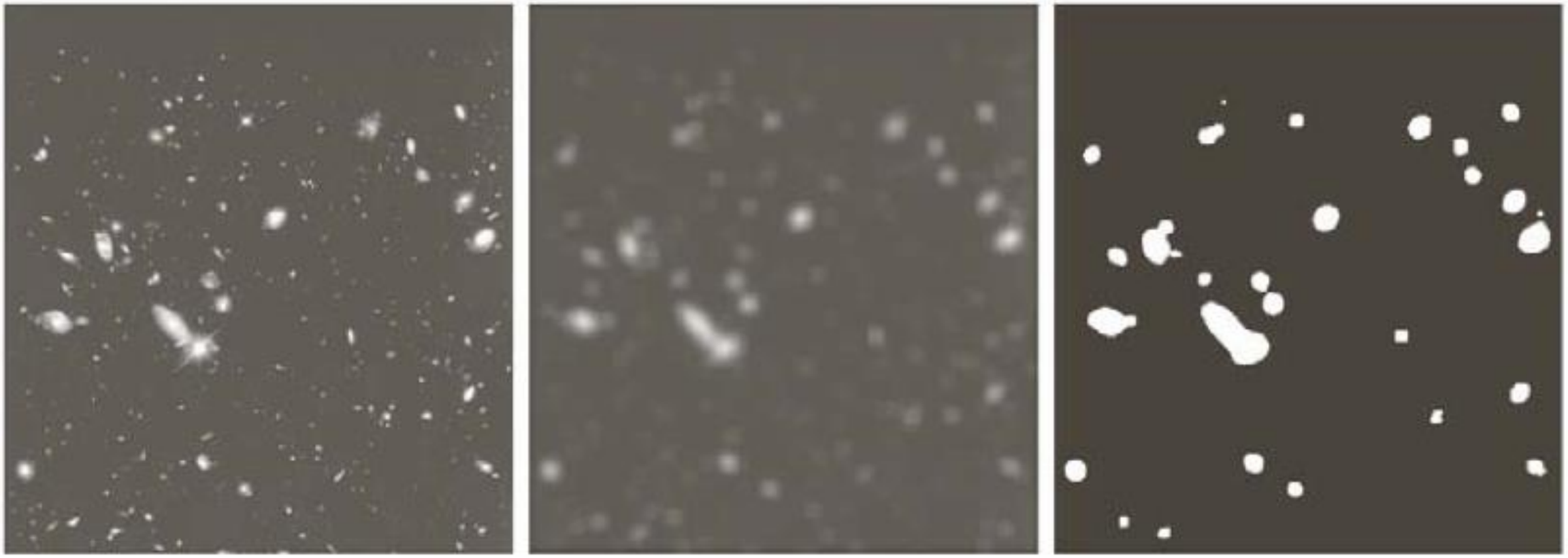
By smoothing the original image we get rid of lots of the finer detail which leaves only the gross features for thresholding



Original Image **Smoothed Image** **Thresholded Image**

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Another Smoothing Example



a b c

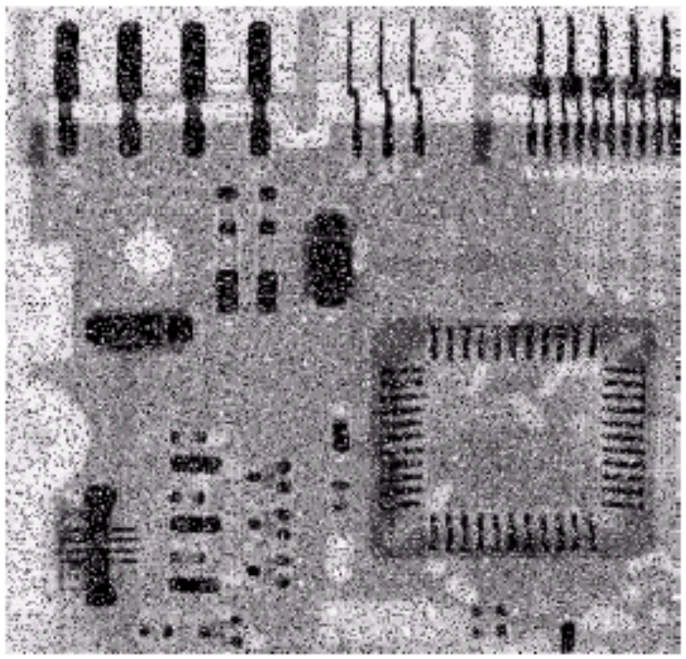
FIGURE 3.34 (a) Image of size 528 × 485 pixels from the Hubble Space Telescope. (b) Image filtered with a 15 × 15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Smoothing Filters

- One problem with the lowpass filter is it blurs edges and other sharp details
- If the intent is to achieve noise reduction, one approach can be to use median filtering
  - The value of each pixel is replaced by the median pixel value in the neighborhood (as opposed to the average)
  - Particularly effective when the noise consists of strong, spike like components and edge sharpness is to be preserved
- The median m of a set of values is such that half of the values are greater than m and half are less than m
- To implement, sort the pixel values in the neighborhood, choose the median and assign this value to the pixel of interest
- Forces pixels with distinct intensities to be more like their neighbors

Averaging Filter Vs. Median Filter Example



Original Image With Noise

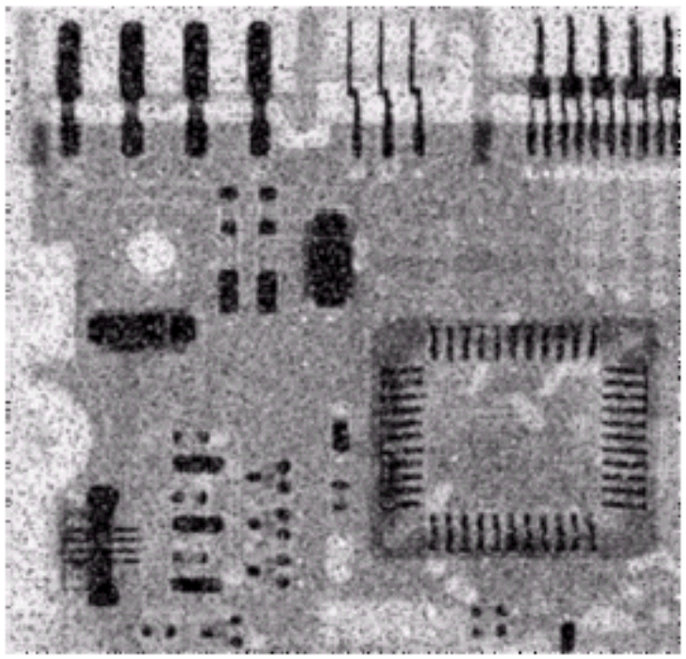


Image After Averaging Filter

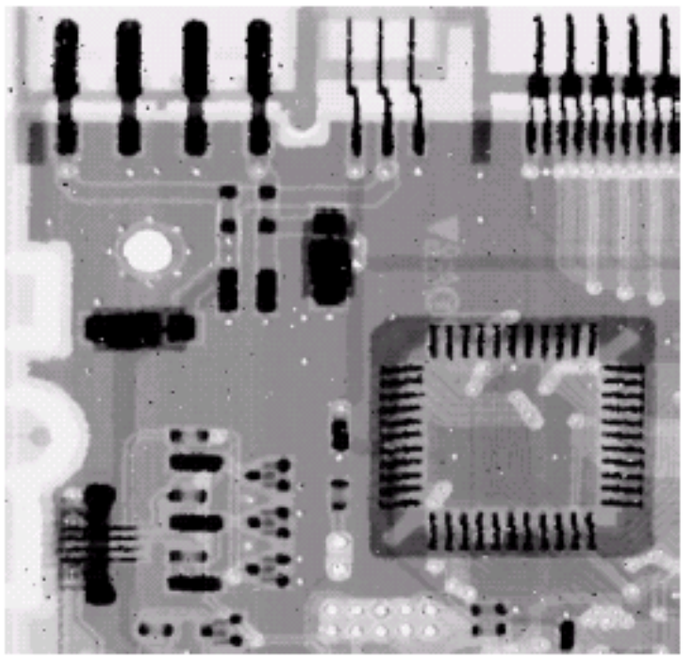


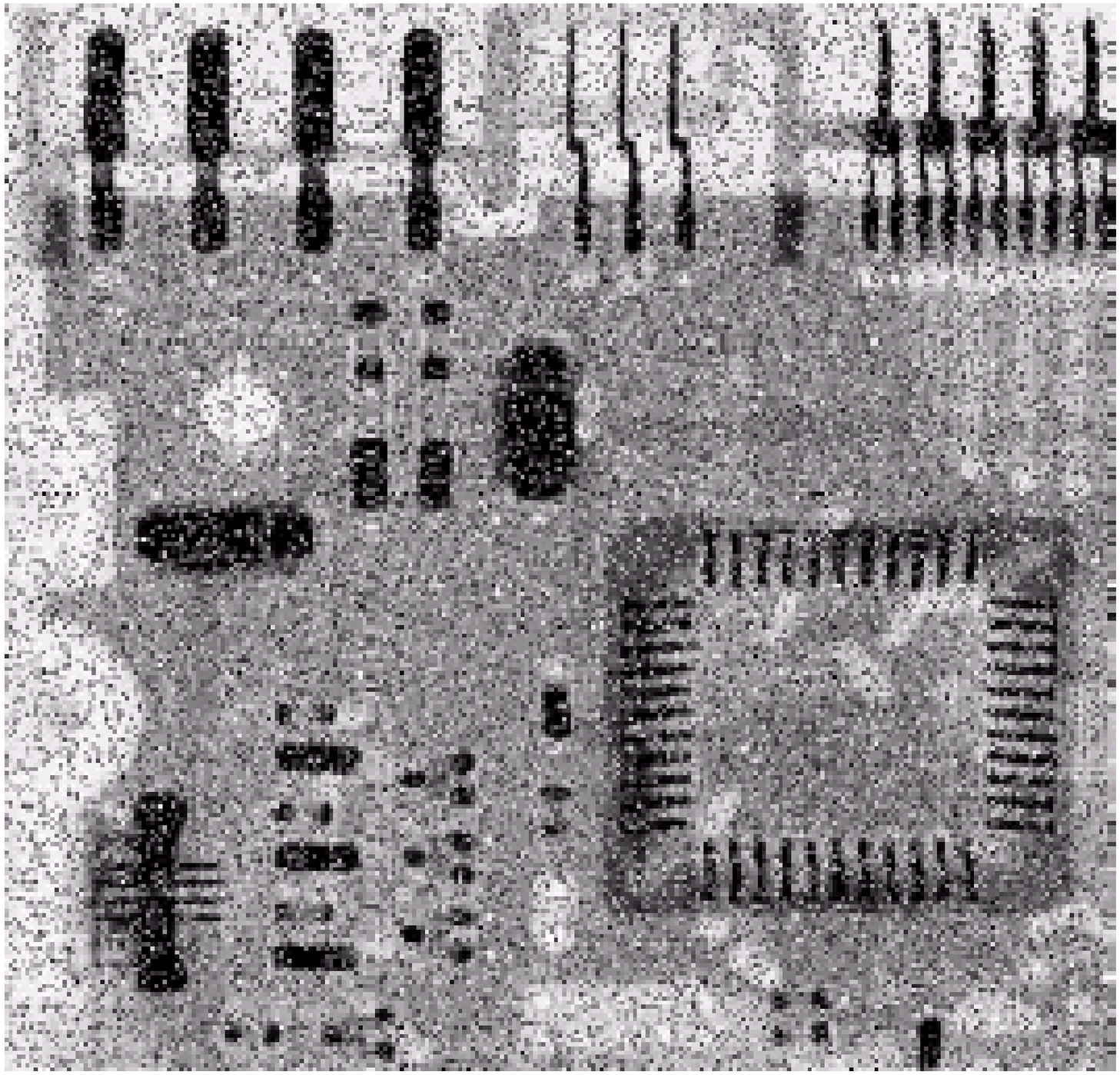
Image After Median Filter

Filtering is often used to remove noise from images

Sometimes a median filter works better than an averaging filter

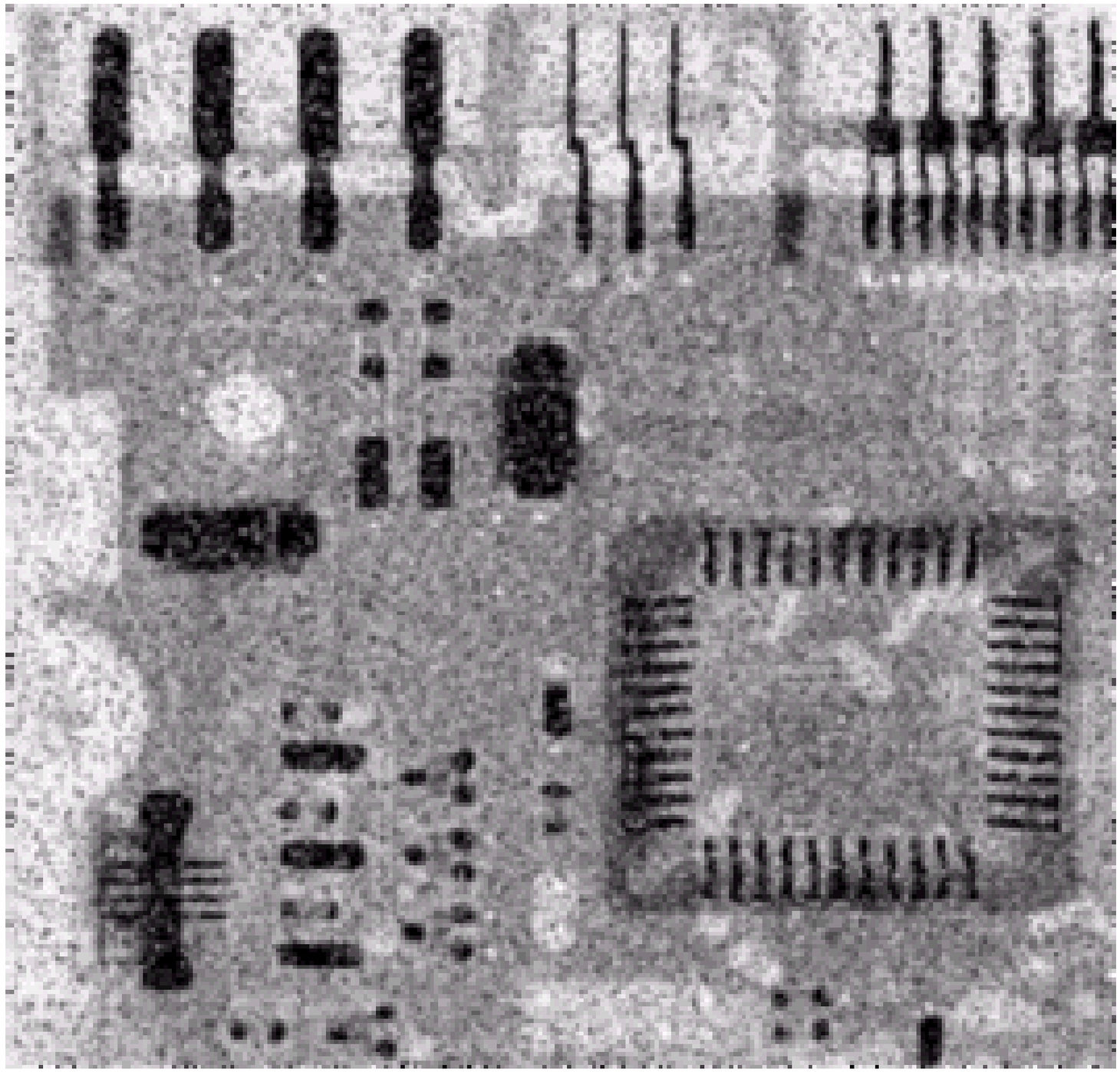
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Averaging Filter Vs. Median Filter Example



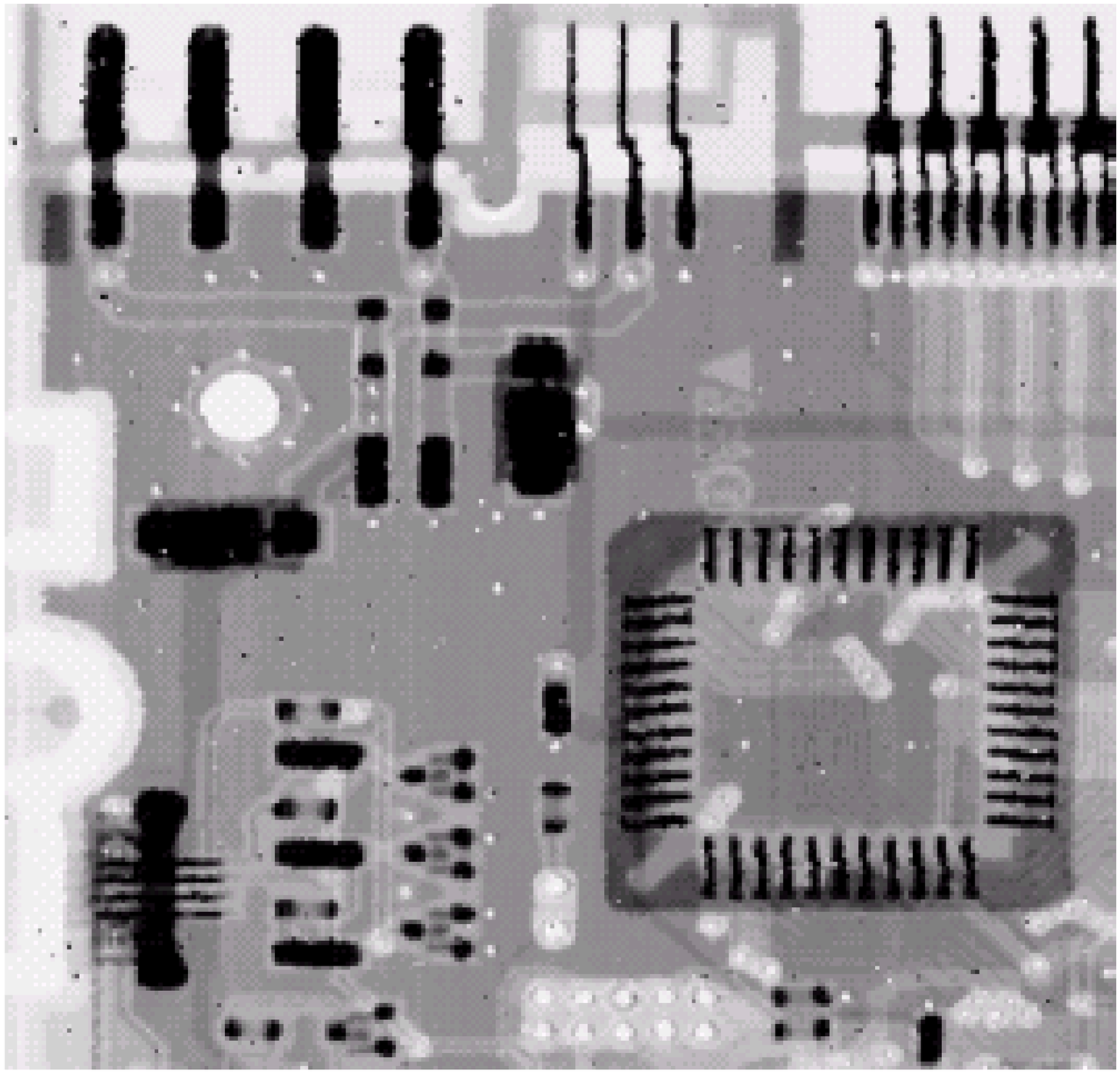
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Averaging Filter Vs. Median Filter Example



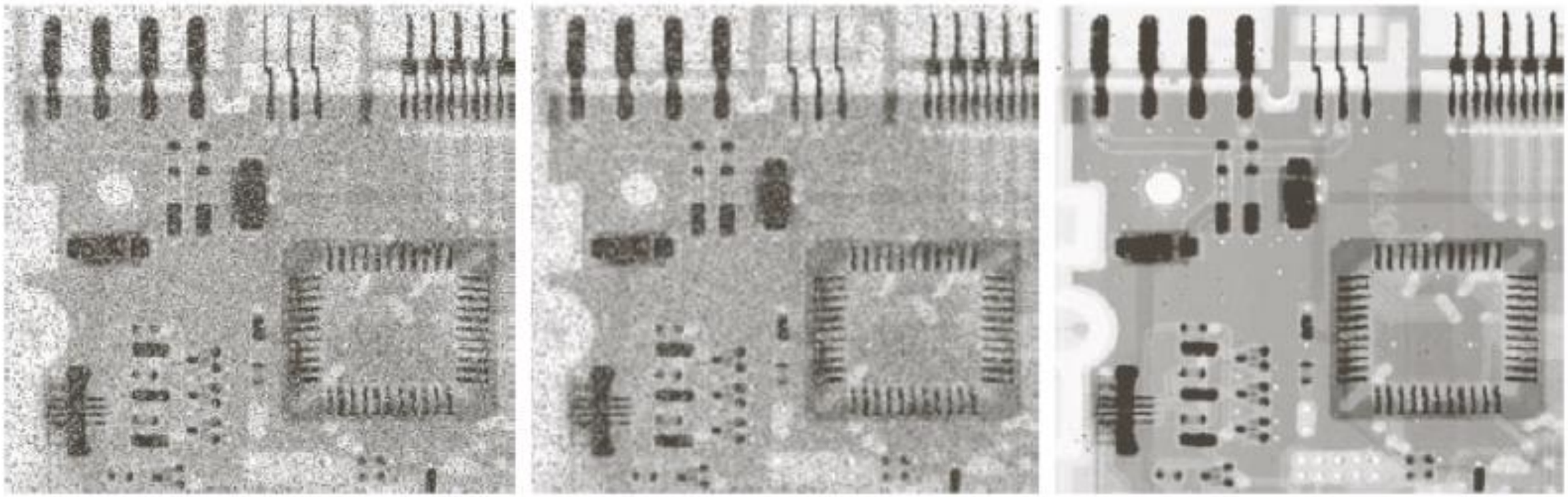
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Averaging Filter Vs. Median Filter Example



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Averaging Filter Vs. Median Filter Example



a b c

FIGURE 3.35 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3×3 averaging mask. (c) Noise reduction with a 3×3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Sharpening Filters (High Pass)

- The shape of the impulse response needed to implement a high-pass (sharpening) filter indicates the filter should have positive coefficients near its center and negative coefficients in the outer periphery
- For a 3x3 mask, the simplest arrangement is to have the center coefficient positive and all others negative

$$\frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Sharpening Filters (High Pass)

- Note the sum of the coefficients is zero

  – When the mask is over a constant or slowly varying region the output is zero or very small

    - This filter eliminates the zero-frequency term

    - Eliminating this term reduces the average gray-level value in the image to zero (will reduce the global contrast of the image)

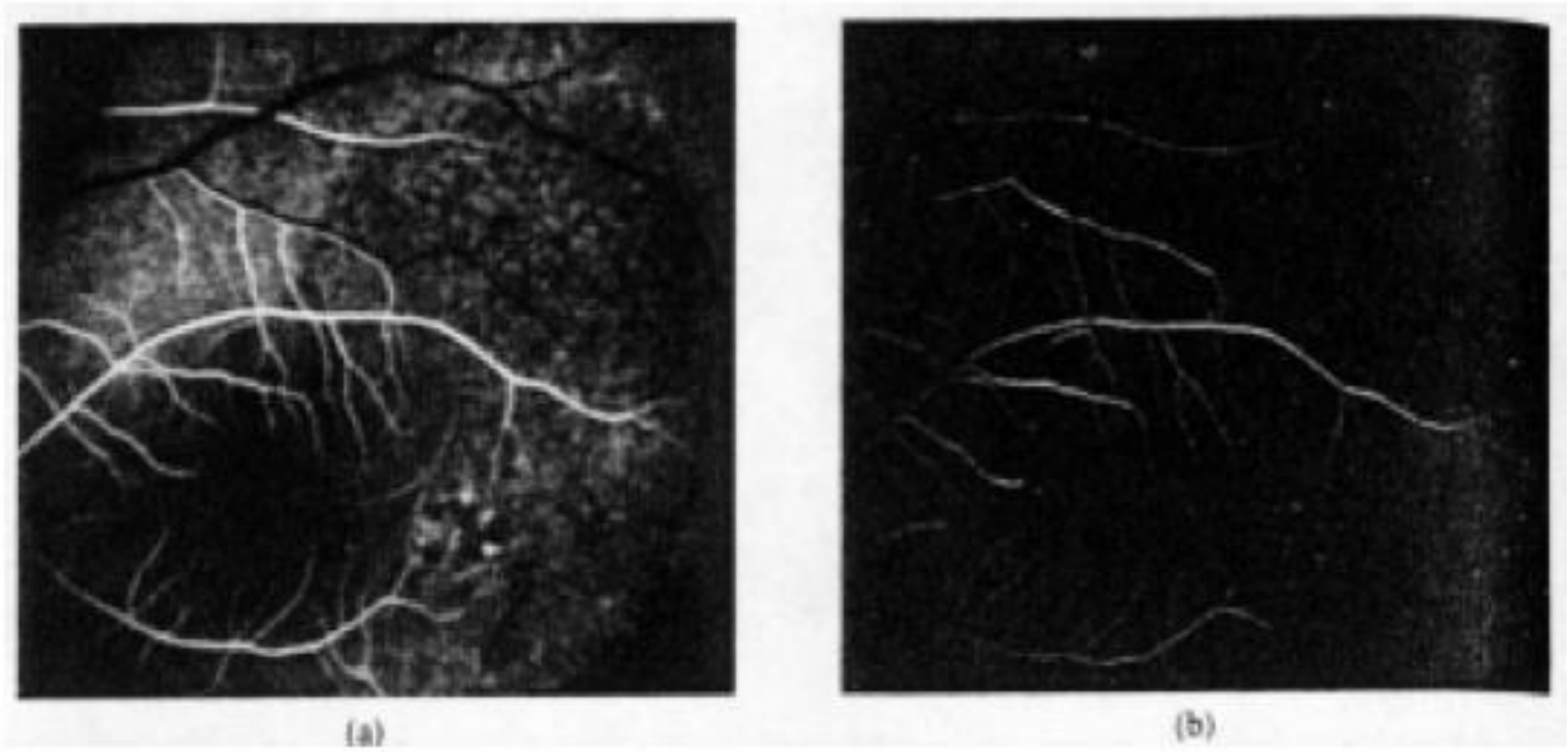
    - Result will be a some what edge-enhanced image over a dark background

- Reducing the average gray-level value to zero implies some negative gray levels

  – The output should be scaled back into an appropriate range

    [0, L-1]

Sharpening Filters (High Pass)



(a) (b)

Sharpening Filters (High-boost)

• A high-pass filter may be computed as:

$$\text{High-pass} = \text{Original} - \text{Lowpass}$$

• Multiplying the original by an amplification factor yields a high-boost or high-frequency-emphasis filter

$$\text{High-boost} = A(\text{Original}) - \text{Lowpass}$$

$$= (A - 1)(\text{Original}) + \text{Original} - \text{Lowpass}$$

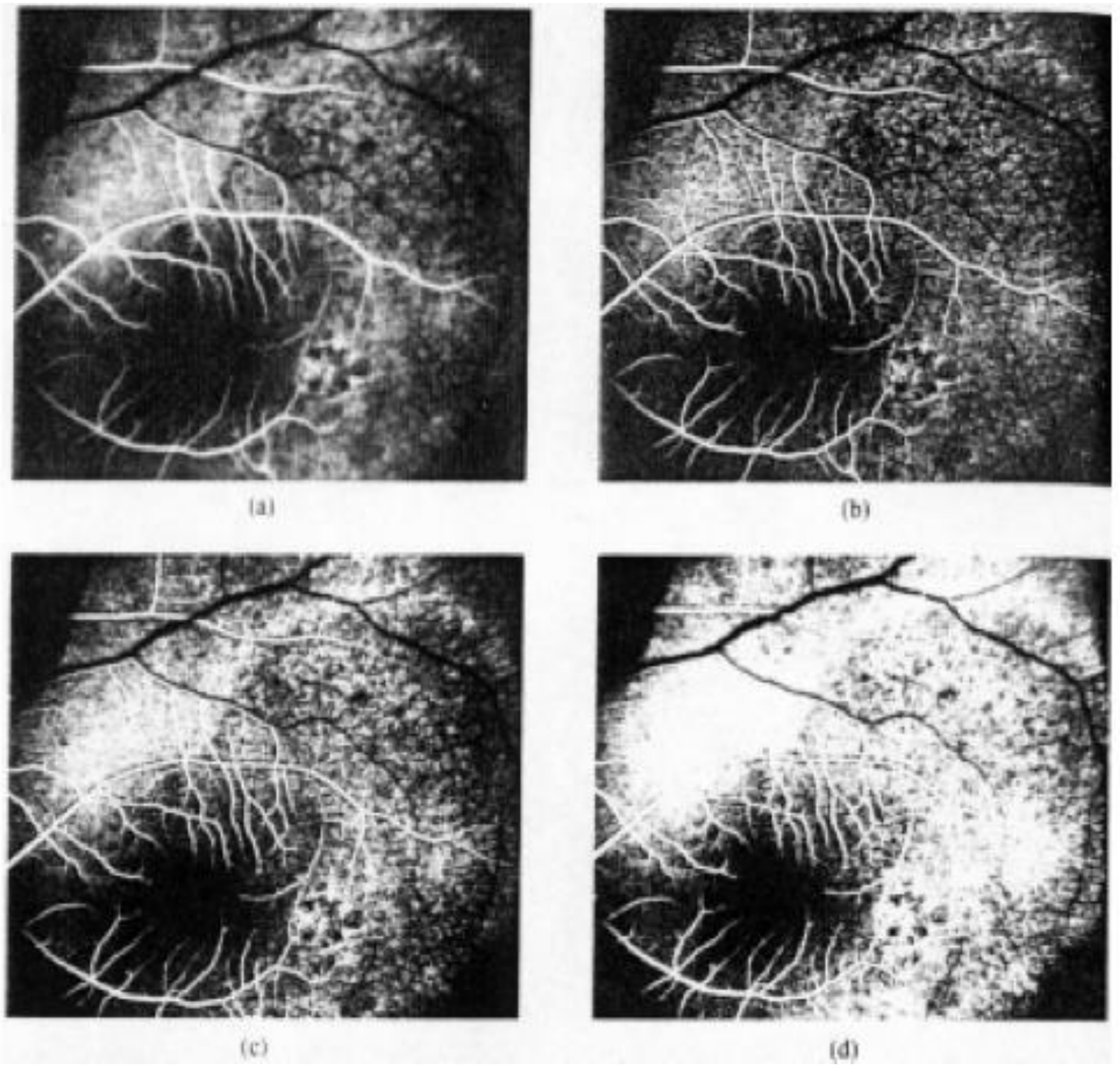
$$= (A - 1)(\text{Original}) + \text{High-pass}$$

– If A>1, part of the original image is added to the high-pass result (partially restoring low frequency components)

– Result looks more like the original image with a relative degree of edge enhancement that depends on the value of A

– May be implemented with the center coefficient value w=9A-1 (A≥1)

Sharpening Filters (High-boost)

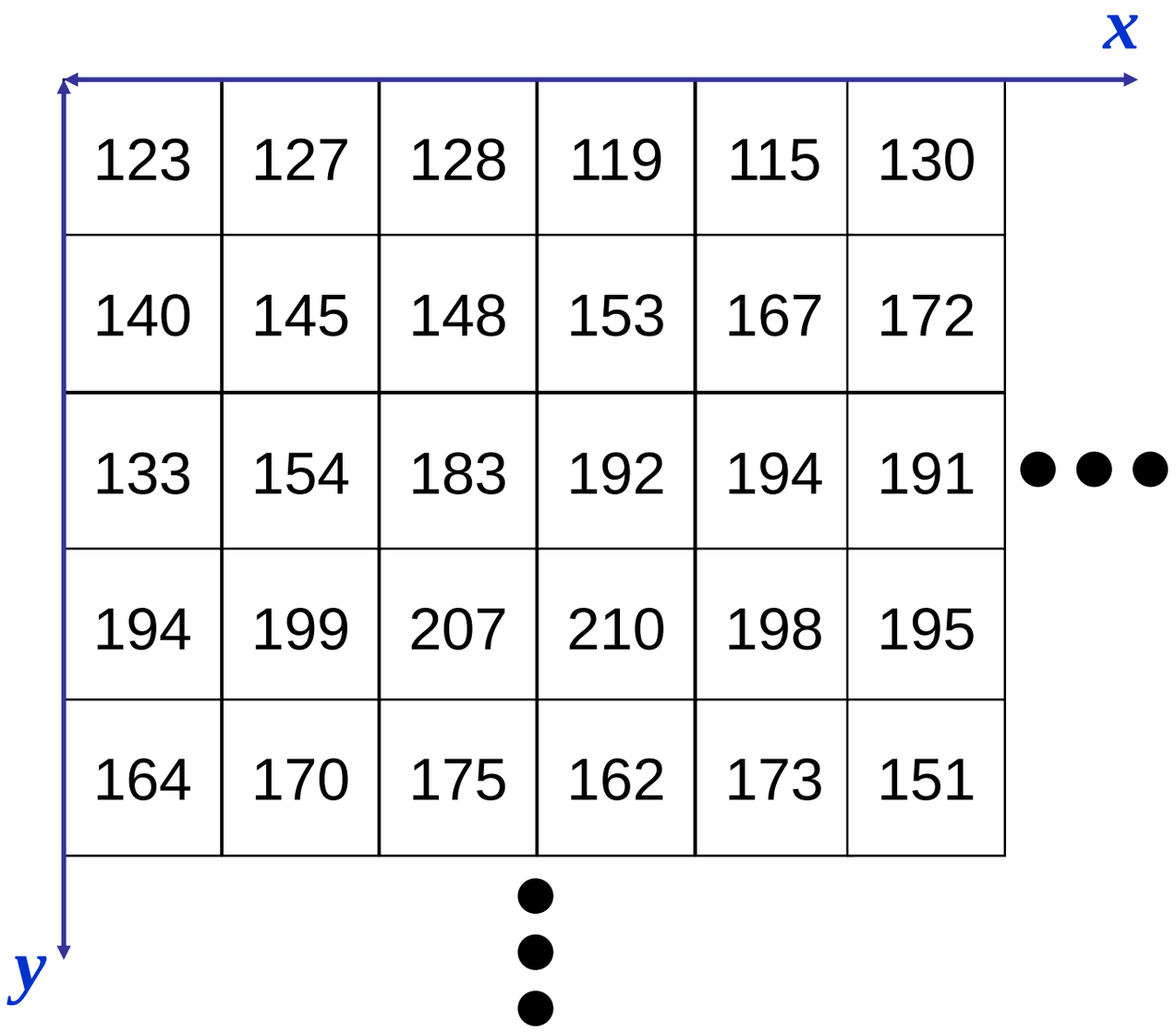


A=1.1

A=1.15

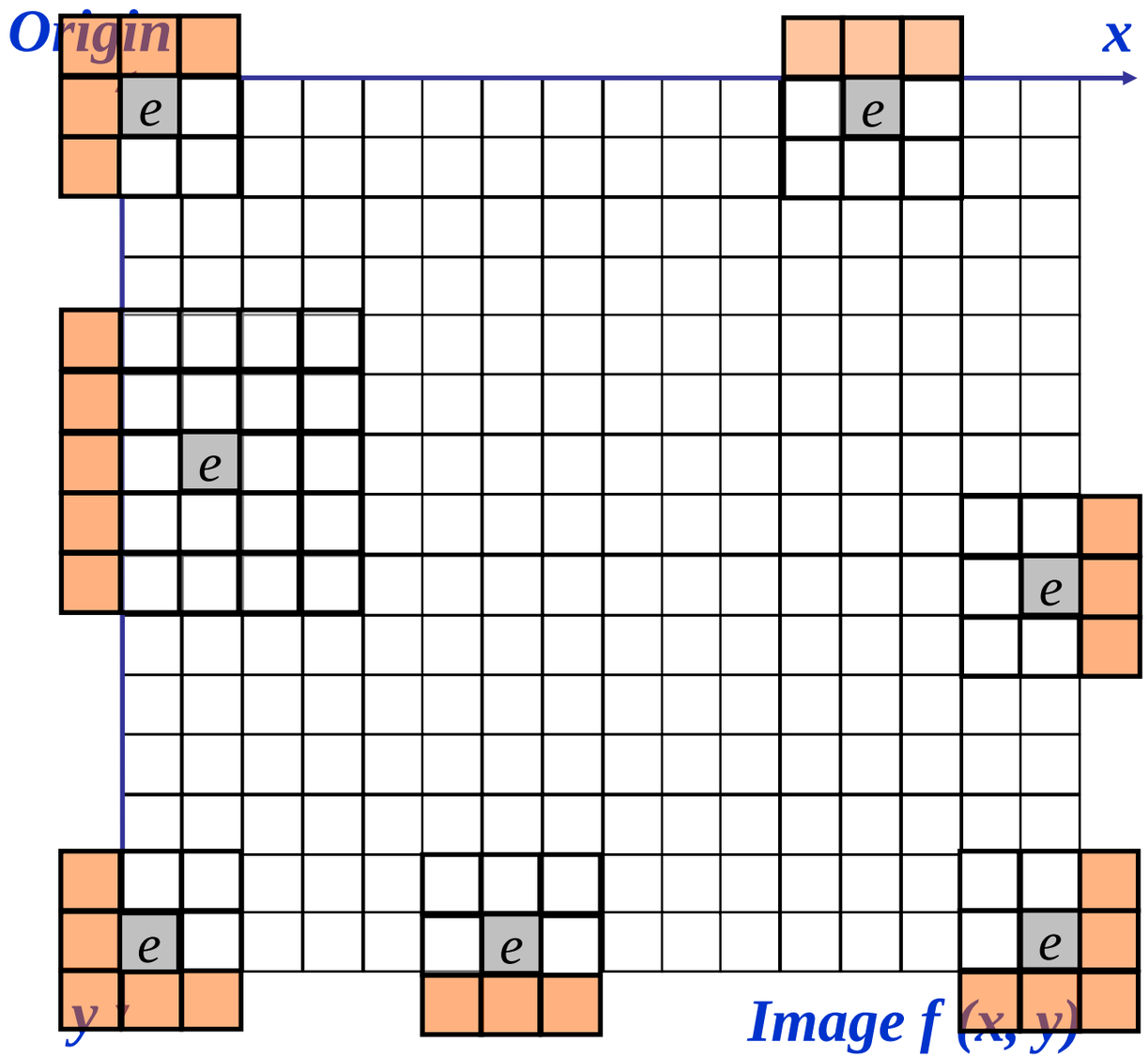
A=1.2

Simple Neighbourhood Operations Example



Strange Things Happen At The Edges

At the edges of an image we are missing pixels to form a neighbourhood



Strange Things Happen At The Edges (cont…)

There are a few approaches to dealing with missing edge pixels:

Omit missing pixels

- Only works with some filters
- Can add extra code and slow down processing

Pad the image

- Typically with either all white or all black pixels

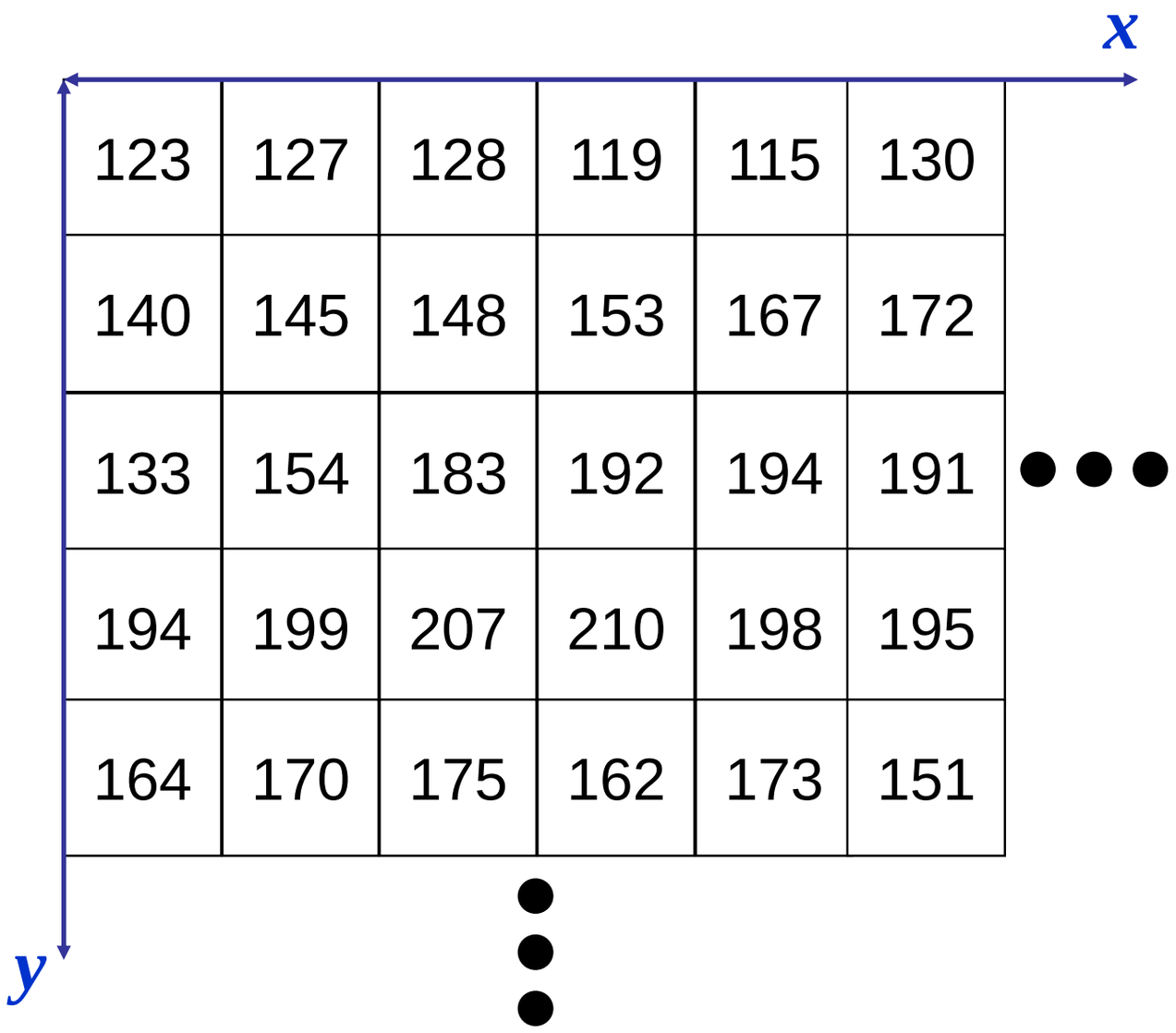
Replicate border pixels

Truncate the image

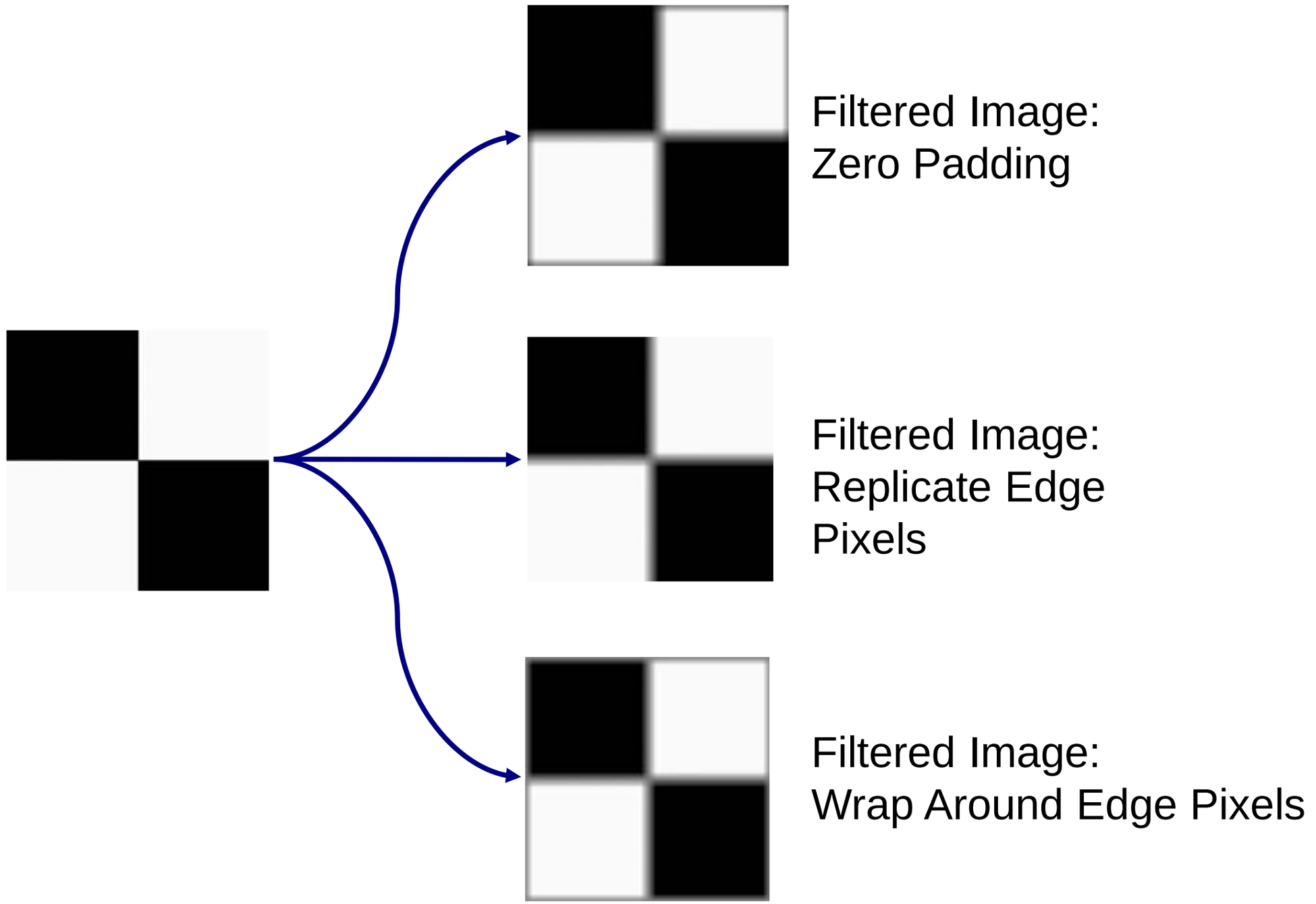
Allow pixels *wrap around* the image

- Can cause some strange image artefacts

Simple Neighbourhood Operations Example

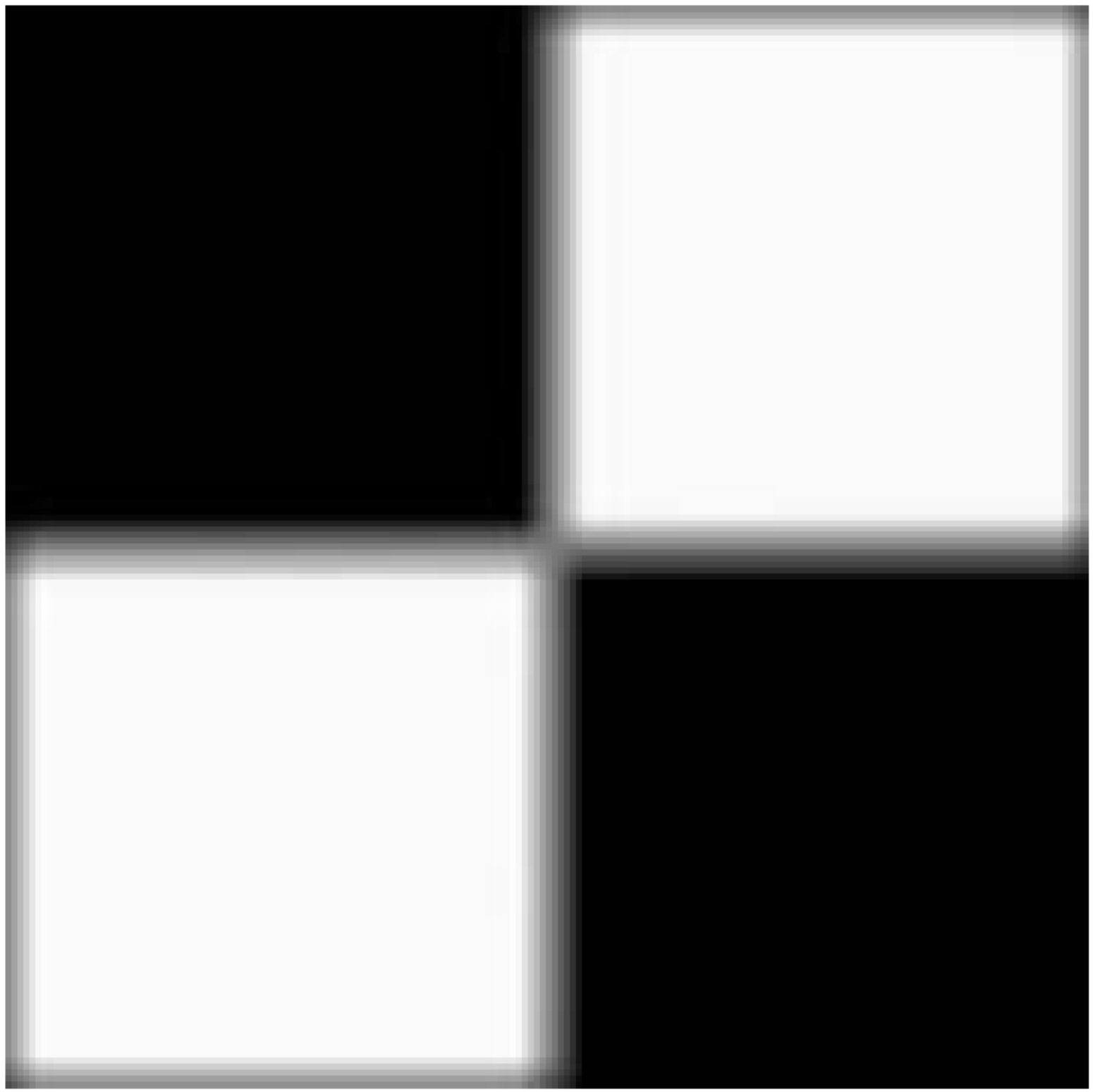


Strange Things Happen At The Edges (cont…)

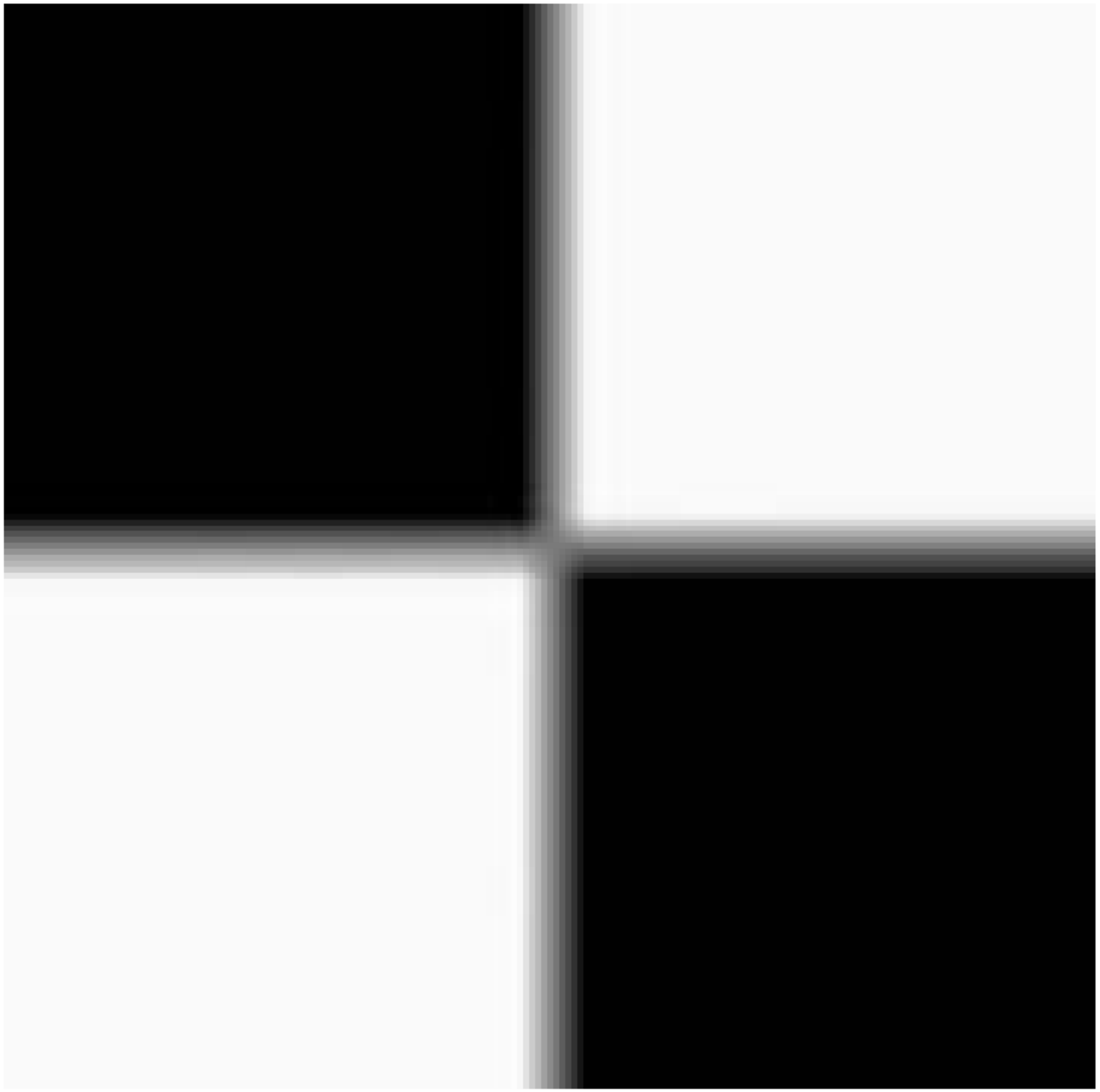


Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Strange Things Happen At The Edges (cont…)



Strange Things Happen At The Edges (cont…)



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Strange Things Happen At The Edges (cont…)

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Correlation & Convolution

The filtering we have been talking about so far is referred to as *correlation* with the filter itself referred to as the ***correlation kernel***

Convolution is a similar operation, with just one subtle difference

z_1	z_2	z_3
z_4	z_4	z_5
z_6	z_7	z_8

Original Image Pixels

$*$

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Filter

$$e_{processed} = z_1*w_1 + z_2*w_{2+} z_3*w_{3+} z_4*w_4 + z_5*w_5 + z_6*w_6 + z_7*w_7 + z_8*w_8 + z_9*w_9$$

For symmetric filters it makes no difference

Correlation & Convolution

The filtering we have been talking about so far is referred to as *correlation* with the filter itself referred to as the *correlation kernel*

Convolution is a similar operation, with just one subtle difference

z_1	z_2	z_3
z_4	z_4	z_5
z_6	z_7	z_8

Original Image Pixels

$*$

w_9	w_8	w_7
w_6	w_5	w_4
w_3	w_2	w_1

Filter

$$e_{processed} = z_1*w_9 + z_2*w_{8+} z_3*w_{7+} z_4*w_6 + z_5*w_5 + z_6*w_4 + z_7*w_3 + z_8*w_2 + z_9*w_1$$

For symmetric filters it makes no difference

Zooming and Shrinking Digital Images

- Zooming is viewed as oversampling and shrinking is viewed as under sampling.
- Zooming is a 2 step process: the creation of new pixel locations and assignment of gray levels to those new locations.
- For example, say we want to zoom an image of size 500 X 500 to 750 X 750.
- We can use nearest neighbor interpolation for zooming.
- Pixel replication is the special case of nearest neighbor interpolation.
- Pixel replication is used to zoom the image by an integer number of times.
- Here new locations are exact duplicates of old locations.
- It is very fast but produces check board effect and hence is undesirable for larger magnification.

Zooming Digital Images

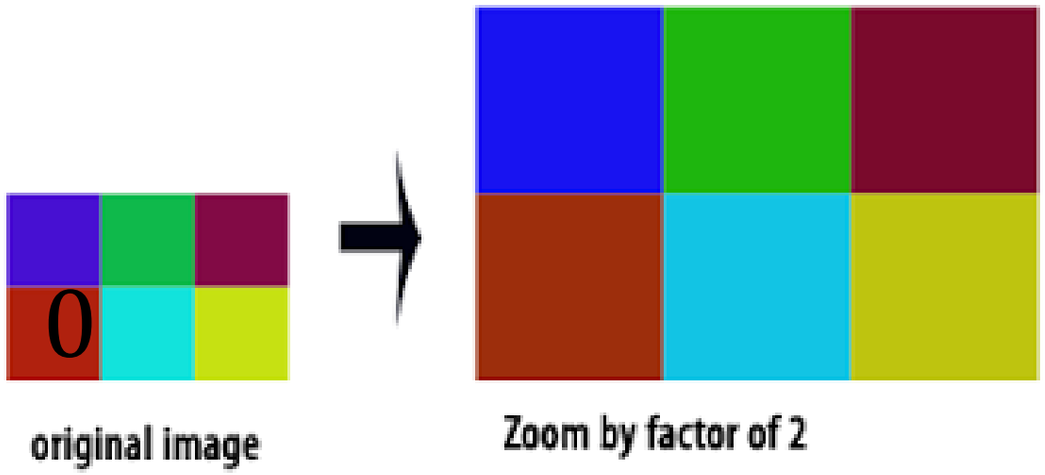
There are many methods of gray level assignments, for examples:

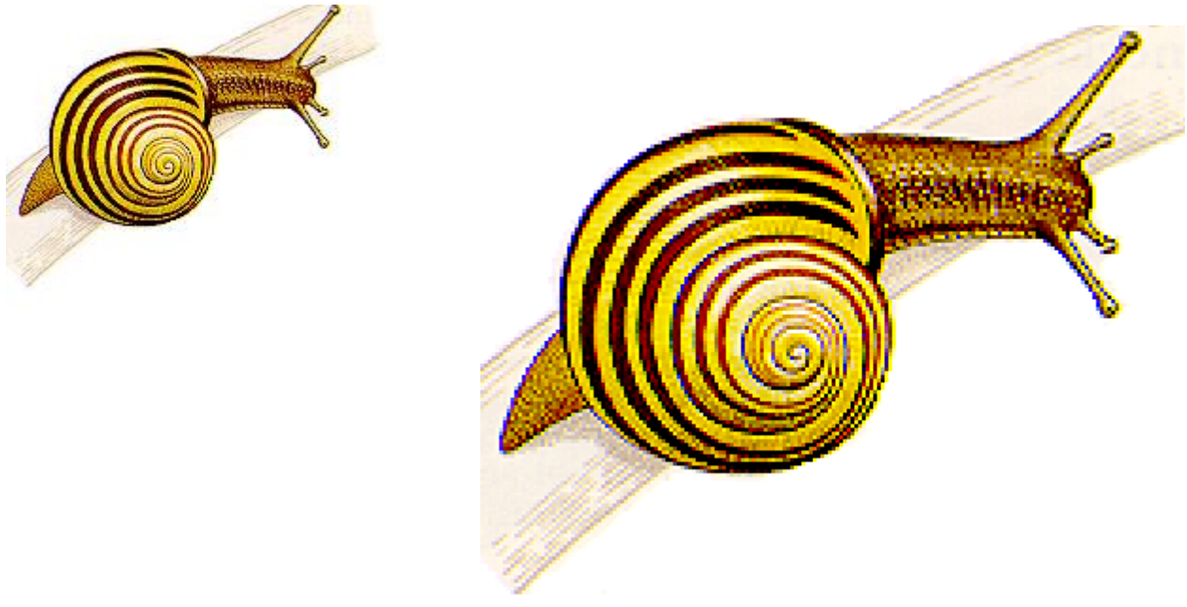
- Nearest neighbor interpolation
- Bilinear interpolation
- K-Times zooming

Image interpolation is used to change the X and Y dimensions of an imag

Magnification by Replication

- Replication is **Zero-Order hold**
- where each pixel along the scan line is repeated once and then each scan line is repeated.
  - i.e column and row duplication
  - Converts M x N image into 2M x 2N image
  - thus creating a blocky effect as in the following figure:
  - Also considered as nearest neighborhood interpolation



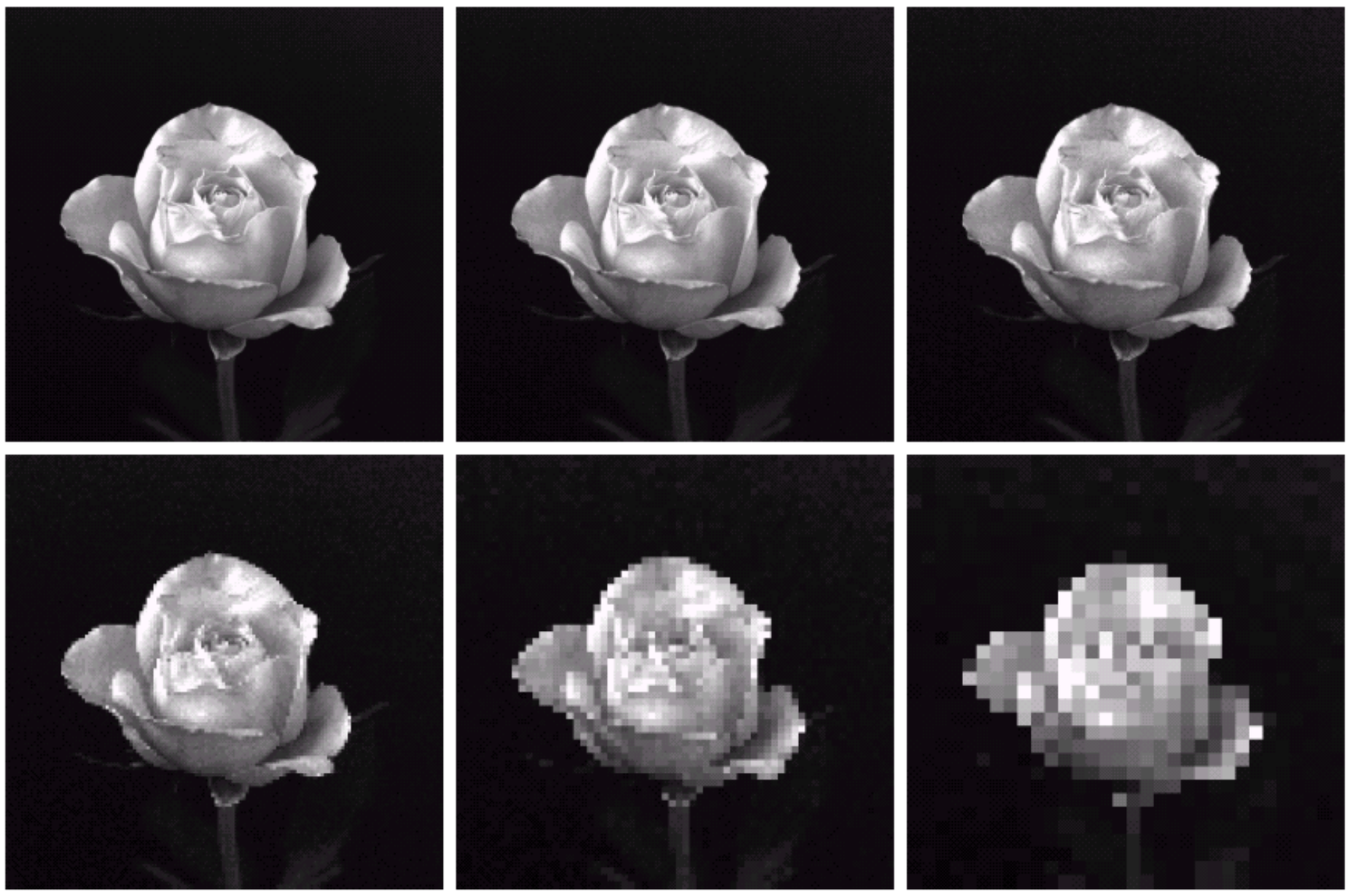


Example

- $\begin{bmatrix} 8 & 4 & 8 \\ 4 & 8 & 4 \\ 8 & 2 & 8 \end{bmatrix}$
- **column duplication**
- $\begin{bmatrix} 8 & 8 & 4 & 4 & 8 & 8 \\ 4 & 4 & 8 & 8 & 4 & 4 \\ 8 & 8 & 2 & 2 & 8 & 8 \end{bmatrix}$
- **Row duplication**
- $\begin{bmatrix} 8 & 8 & 4 & 4 & 8 & 8 \\ 8 & 8 & 4 & 4 & 8 & 8 \\ 4 & 4 & 8 & 8 & 4 & 4 \\ 4 & 4 & 8 & 8 & 4 & 4 \\ 8 & 8 & 2 & 2 & 8 & 8 \\ 8 & 8 & 2 & 2 & 8 & 8 \end{bmatrix}$

Magnification by Replication

- Subsampled and resampled



a b c
d e f

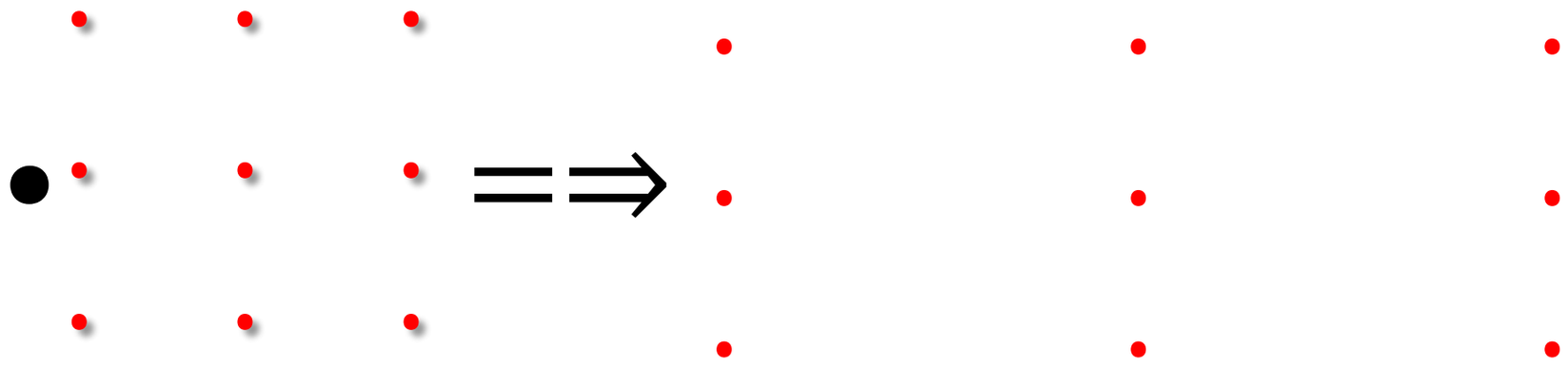
FIGURE 2.20 (a) 1024 × 1024, 8-bit image. (b) 512 × 512 image resampled into 1024 × 1024 pixels by row and column duplication. (c) through (f) 256 × 256, 128 × 128, 64 × 64, and 32 × 32 images resampled into 1024 × 1024 pixels.

Nearest Neighbor Interpolation

- Nearest neighbor interpolation is the simplest method and basically makes the pixels bigger.
- The color of a pixel in the new image is the color of the nearest pixel of the original image.
- If you enlarge 200%, one pixel will be enlarged to a 2 x 2 area of 4 pixels with the same color as the original pixel.
- **Most image viewing and editing software use this type of interpolation to enlarge a digital image for the purpose of closer examination because it does not change the color information of the image and does not introduce any anti-aliasic.**
- For the same reason, it is not suitable to enlarge photographic images because it increases the visibility of jaggies.

Bilinear Interpolation

- Bilinear Interpolation is **First-Order Hold**
- **performed by finding linear interpolation between a adjacent pixels, i.e., finding the average value between two pixels and use that as the pixel value between those two**
- At first straight line is first fitted in between pixels along the row. Then pixels along each column is interpolated. **zero interlace is done in column and average is calculated**



- **m*n**

$$\left.\begin{aligned} v_1(m, 2n) &= u(m, n), \\ &0 \le m \le M-1, 0 \le n \le N-1 \end{aligned}\right\}$$

m*2n

$$\left.\begin{aligned} v_1(m, 2n+1) &= \tfrac{1}{2}[u(m, n) + u(m, n+1)], \\ &0 \le m \le M-1, 0 \le n \le N-1 \end{aligned}\right\}$$

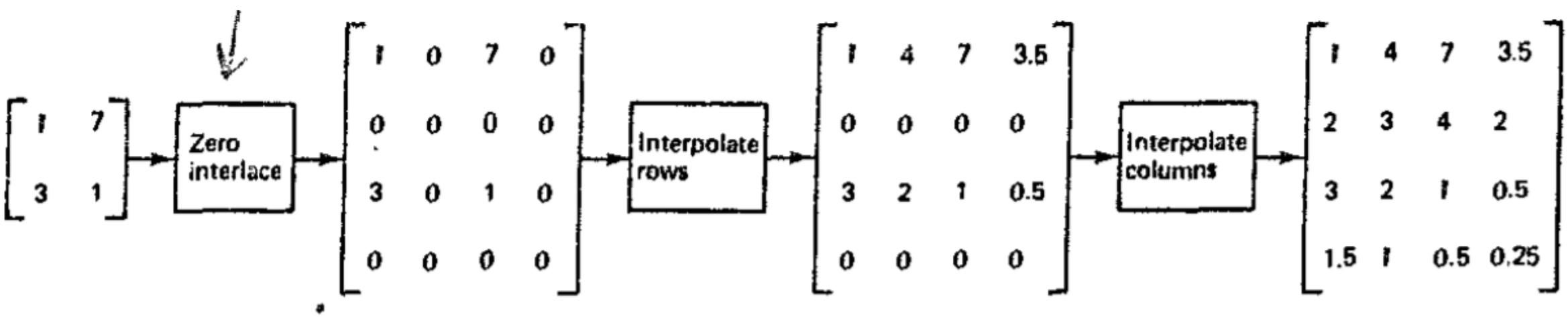
Bilinear Interpolation

- Then straight line is fitted in between pixels along the column. Then pixels along each row is interpolated.
- **zero interlace is done in row and average is calculated**

Linear interpolation of the preceding along columns gives the first result as

$$\left.\begin{aligned} v(2m, n) &= v_1(m, n) \\ v(2m+1, n) &= \tfrac{1}{2}[v_1(m, n) + v_1(m+1, n)], \\ & \quad 0 \le m \le M-1, 0 \le N \le 2N-1 \end{aligned}\right\}$$

- The averaging has an anti-aliasing effect and therefore produces relatively smooth edges with hardly any jaggies.



Example: Magnification by Interpolation

- $\begin{bmatrix} 8 & 4 & 8 \\ 4 & 8 & 4 \\ 8 & 2 & 8 \end{bmatrix}$
- **zero interlace is done in column and average is calculated as (8+4)/2**
- $\begin{bmatrix} 8 & 0 & 4 & 0 & 8 & 0 \\ 4 & 0 & 8 & 0 & 4 & 0 \\ 8 & 0 & 2 & 0 & 8 & 0 \end{bmatrix} \Longrightarrow \begin{bmatrix} 8 & 6 & 4 & 6 & 8 & 4 \\ 4 & 6 & 8 & 6 & 4 & 2 \\ 8 & 5 & 2 & 5 & 8 & 4 \end{bmatrix}$
- **zero interlace is done in row and average is calculated as (8+4)/2**
- $\begin{bmatrix} 8 & 6 & 4 & 6 & 8 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 6 & 8 & 6 & 4 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 8 & 5 & 2 & 5 & 8 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \Longrightarrow \begin{bmatrix} 8 & 6 & 4 & 6 & 8 & 4 \\ 6 & 6 & 6 & 6 & 6 & 3 \\ 4 & 6 & 8 & 6 & 4 & 2 \\ 6 & 5.5 & 5 & 5.5 & 6 & 3 \\ 8 & 5 & 2 & 5 & 8 & 4 \\ 4 & 2.5 & 1 & 2.5 & 4 & 2 \end{bmatrix}$

Bicubic interpolation

- Bicubic interpolation is more sophisticated and produces smoother edges than bilinear interpolation.
- Here, a new pixel is a bicubic function using 16 pixels in the nearest 4 x 4 neighborhood of the pixel in the original image.
- This is the method most commonly used by image editing software, printer drivers and many digital cameras for re-sampling images.

Magnification results

Figure : Gray level interpolation by using the nearest neighbor in case of zooming.

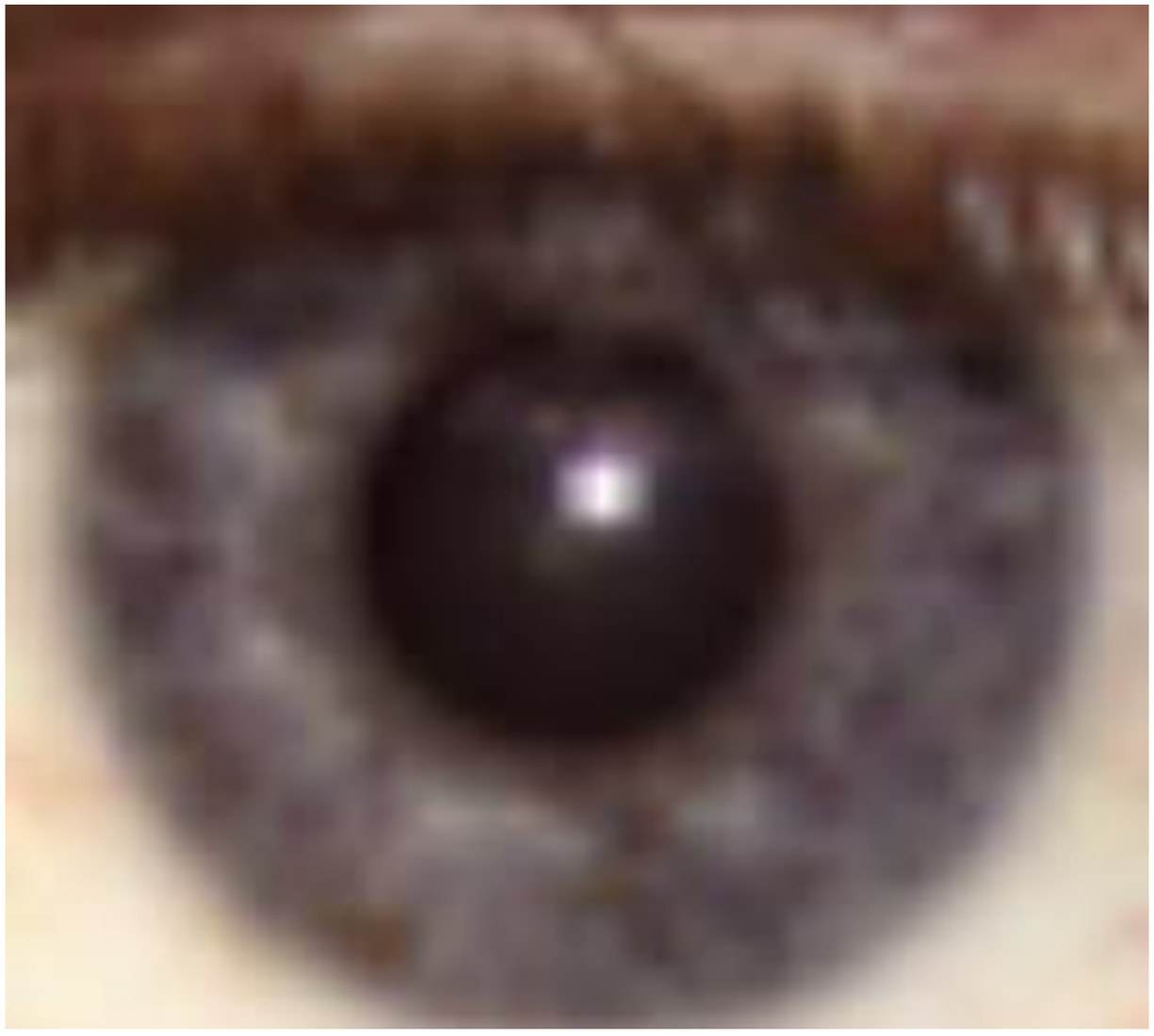
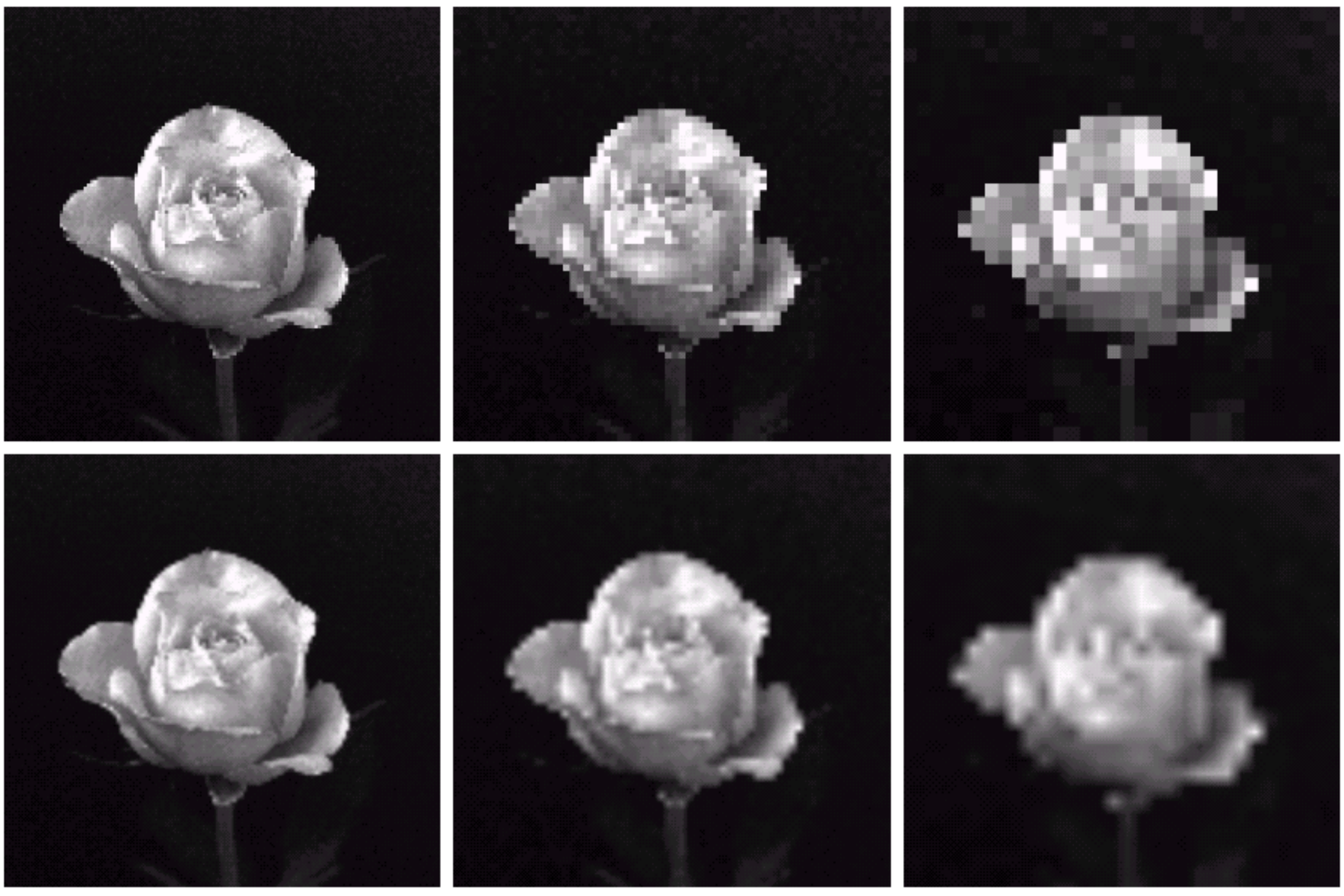


Figure : Gray level interpolation by using the bilinear method in case of zooming.

Magnification results



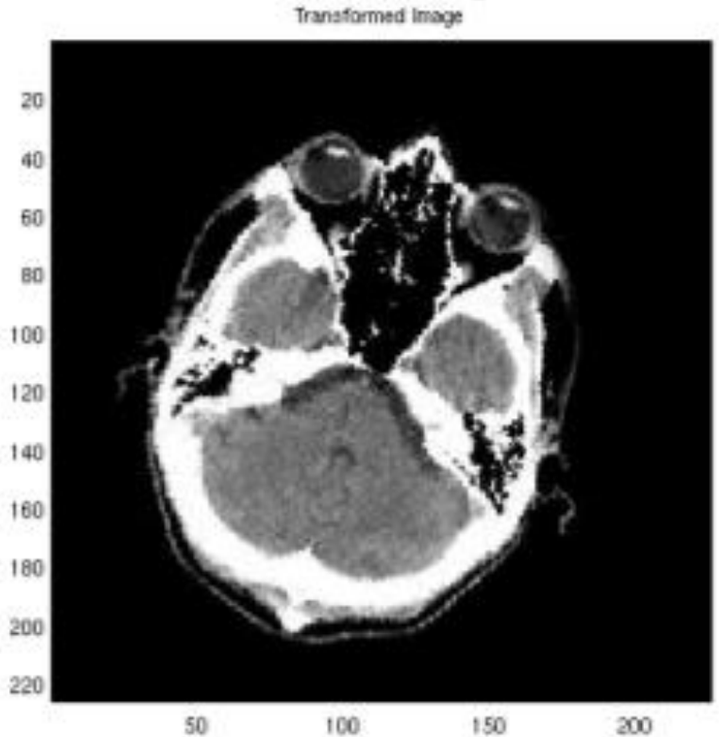
a b c
d e f

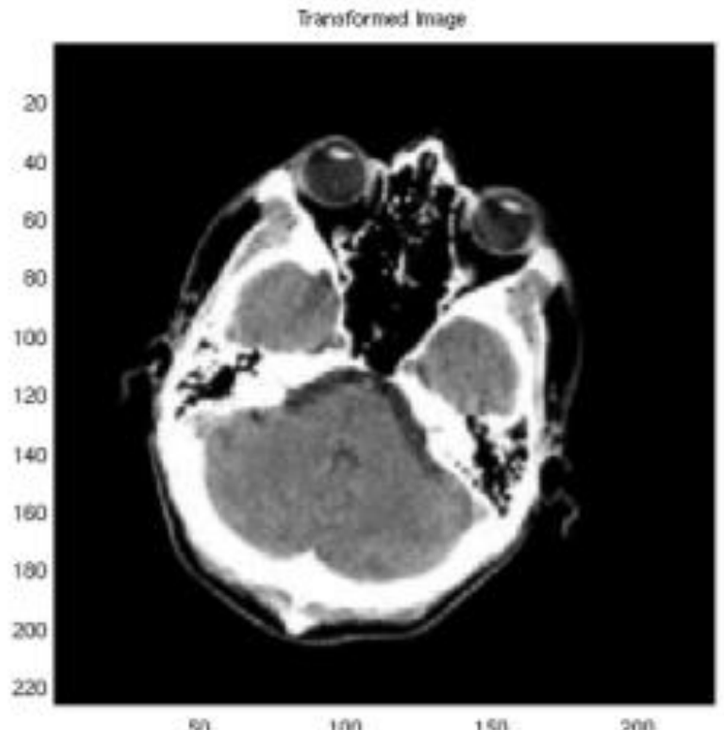
FIGURE 2.25 Top row: images zoomed from 128 × 128, 64 × 64, and 32 × 32 pixels to 1024 × 1024 pixels, using nearest neighbor gray-level interpolation. Bottom row: same sequence, but using bilinear interpolation.

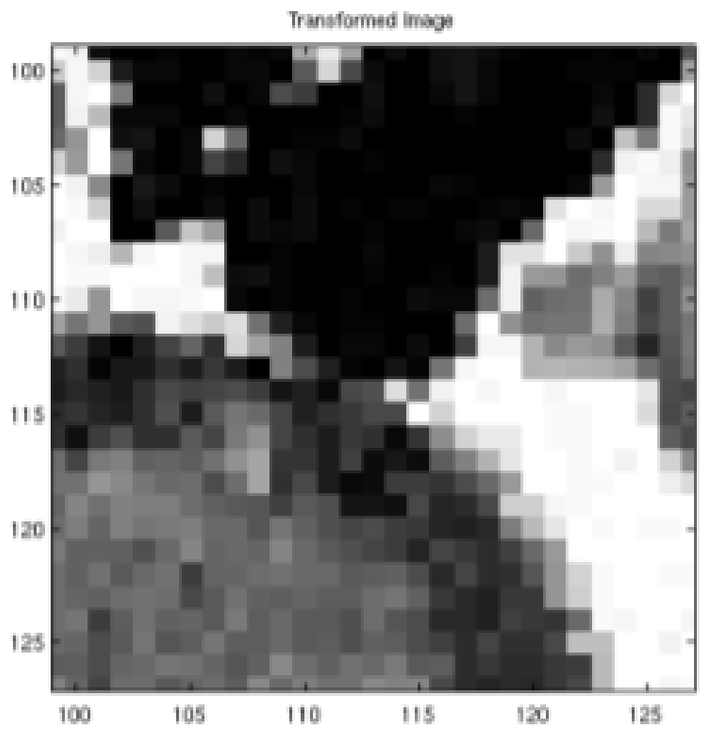
Magnification results

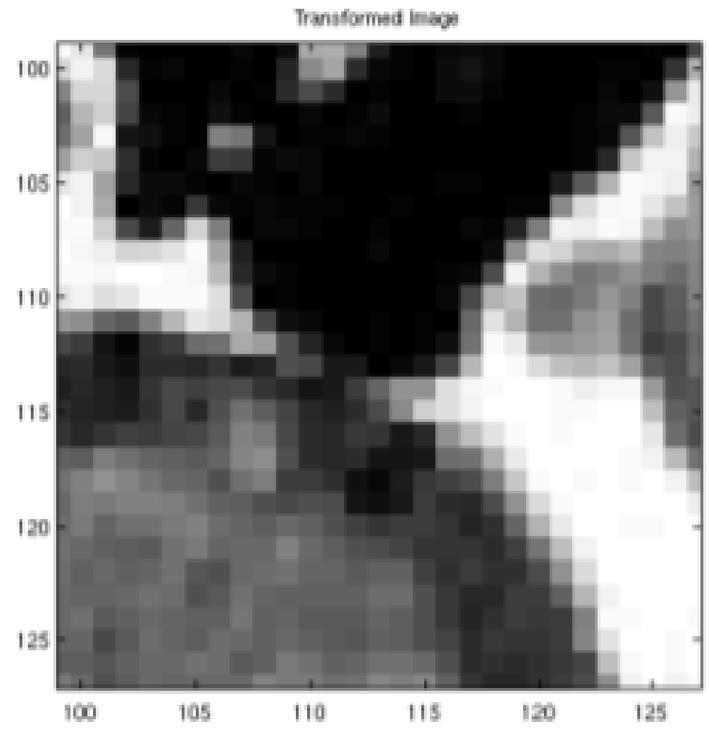
Comparison between Nearest neighbours and bilinear interpolation

- need to zoom it to see see the effect of the interpolation.
- As a influence of the interpolation, we can see that very white pixels of the bone actually appear gray.
- the jagged edges have been significantly improved.









Magnification results

Original Image

Result after replication

Result after Interpolation

K-Times Zooming

- you have to take two adjacent pixels as you did in the zooming twice. Then you have to subtract the smaller from the greater one. We call this output (OP).

- Divide the output(OP) with the zooming factor(K). Now you have to add the result to the smaller value and put the result in between those two values.

- Add the value OP again to the value you just put and place it again next to the previous putted value. You have to do it till you place k-1 values in it.

- Repeat the same step for all the rows and the columns , and you get a zoomed images.

K-Times Zooming

- Suppose you have an image of 2 rows and 3 columns , which is given below. And you have to zoom it thrice or three times.
- K in this case is 3. K = 3.

The number of values that should be inserted is k-1 = 3-1 = 2.

15	30	15
30	15	30

K-Times Zooming

Row wise zooming

Take the first two adjacent pixels. Which are 15 and 30.
Subtract 15 from 30. 30-15 = 15.
Divide 15 by k. 15/k = 15/3 = 5. We call it OP.(where op is just a name)
Add OP to lower number. 15 + OP = 15 + 5 = 20.
Add OP to 20 again. 20 + OP = 20 + 5 = 25.
We do that 2 times because we have to insert k-1 values.
Now repeat this step for the next two adjacent pixels. It is shown in the first table.
After inserting the values , you have to sort the inserted values in ascending order, so there remains a symmetry between them.
It is shown in the second table

15	30	15
20	25	20
25	20	25
30	15	30

K-Times Zooming

Column wise zooming

The same procedure has to be performed column wise. The procedure include taking the two adjacent pixel values, and then subtracting the smaller from the bigger one. Then after that , you have to divide it by k. Store the result as OP. Add OP to smaller one, and then again add OP to the value that comes in first addition of OP. Insert the new values.

Here what you got after all that

15	20	25	30	25	20	15
20	21	22	25	22	21	20
25	22	21	20	21	22	25
30	25	20	15	20	25	30

15	**30**	**15**
30	**15**	**30**

K-Times Zooming

New image size

The best way to calculate the formula for the dimensions of a new image is to compare the dimensions of the original image and the final image. The dimensions of the original image were 2 X 3. And the dimensions of the new image are 4 x 7.

The formula thus is:

(K (number of rows minus 1) + 1) X (K (number of cols minus 1) + 1)

Advantages and disadvantages

The one of the clear advantage that k time zooming algorithm has that it is able to compute zoom of any factor which was the power of pixel replication algorithm , also it gives improved result (less blurry) which was the power of zero order hold method. So hence It comprises the power of the two algorithms.

The only difficulty this algorithm has that it has to be sort in the end , which is an additional step , and thus increases the cost of computation.

Shrinking Images

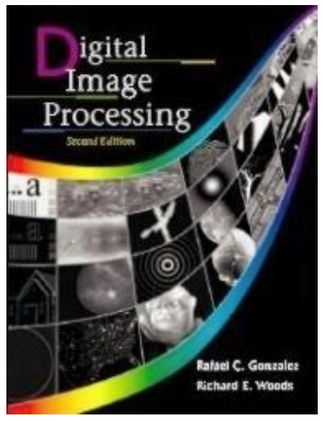
Shrinking may be viewed as undersampling. Image shrinking is performed by **row-column deletion**. For example, to shrink an image by one-half, we delete every other row and column.

$$\begin{bmatrix} 69 & 69 & 50 & 50 & 80 & 80 \\ 69 & 69 & 50 & 50 & 80 & 80 \\ 45 & 45 & 60 & 60 & 66 & 66 \\ 45 & 45 & 60 & 60 & 66 & 66 \\ 30 & 30 & 55 & 55 & 80 & 80 \\ 30 & 30 & 55 & 55 & 80 & 80 \end{bmatrix} = \begin{bmatrix} 69 & 69 & 50 & 50 & 80 & 80 \\ 45 & 45 & 60 & 60 & 66 & 66 \\ 30 & 30 & 55 & 55 & 80 & 80 \end{bmatrix} = \begin{bmatrix} 69 & 50 & 80 \\ 45 & 60 & 66 \\ 30 & 55 & 80 \end{bmatrix}$$

Original image — image with rows deleted — image with rows and columns deleted

Shrinking, in the other hand involves reduction of pixels and it means lost of irrecoverable information. In this case scaling algorithm is to find the right pixels to throw away.

References



"Digital Image Processing", Rafael C. Gonzalez & Richard E. Woods, Addison-Wesley, 2002

–"Fundamentals of Digital Image Processing" Anil K. Jain, 1989

–Image Processing and Pattern Recognition Slides of Dr. Sanjeeb Prasad Panday

–https://www.slideshare.net/nidhalelabbadi/image-processing-zooming-and-shrinking

Thank you