Outline

- Image Enhancement in the Spatial Domain (Cont.)
  - Spatial Filtering: Introduction
  - Smoothing filters
  - Sharpening filters
  - Hybrid Enhancement
- Image Enhancement in the Frequency Domain
  - Filtering in the frequency domain
  - Relationship with spatial processing
  - Smoothing in the frequency domain

Spatial Filtering: Introduction

- Filtering in the spatial domain is moving the filter over the image pixel by pixel, and calculating the output for each pixel using an operation between the values of the image and the values of the filter
- For a general mask this operation can be any function (linear or non-linear) of the image pixels and filter pixels
- For a linear filter the operation is restricted to sum of products of image and the filter values

$$R(x,y) = \sum_{x}^{'} \sum_{y}^{'} f(x+x', y+y') \times w(x', y')$$

where $f(x,y)$ is the image, $w(x,y)$ the mask, x' and y' depend on the size of the filter usually symmetric around zero

Spatial Filtering: Introduction

- Similar to the convolution operation
- Same equation can be written using a single summation by shaping the matrix into vectors

$$R(x, y) = \sum_i w_i z_i,$$

where z_i is the part of the image overlapping with the filter

Spatial Filtering: Introduction (Cont.)

- The filtering domain is symmetric around the pixel in focus
- What happens around the borders of the image, where the part of the mask lies outside the image
- Trivial solutions can be used, not crucial since it affects only a minor part of the image
  - Use parts of the mask that lies on the image
  - Ignore the border pixels, yield a smaller filtered image

Linear Smoothing Filters

- Smoothing filters result in a blurred version of the input image
- The sharp transitions are reduced
  - Good for noise reduction since noise is typically in the form a sharp transition in pixel values
  - Bad for edges since edges are also sharp transitions but they are not noise!
- Smoothing filters yield blurring by simply averaging the pixel values of the image
- The general linear form includes weights for different pixels

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

- The numerator performs the smoothing and the denominator is necessary to preserve the overall gray values of the image

Linear Smoothing Filters: Examples

- Free parameters of design are a, b the size of the filter, and the values of filter
- Depending on a certain application, optimum parameters can be selected based on a cost function
- Some common/simple low pass filter examples are

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

Linear Smoothing Filters: Examples (Cont.)

- This filter uses uniform weights hence sometimes an extensive blurring
- To reduce blurring we can assign more weight to pixels close to the center and less weight to pixels further

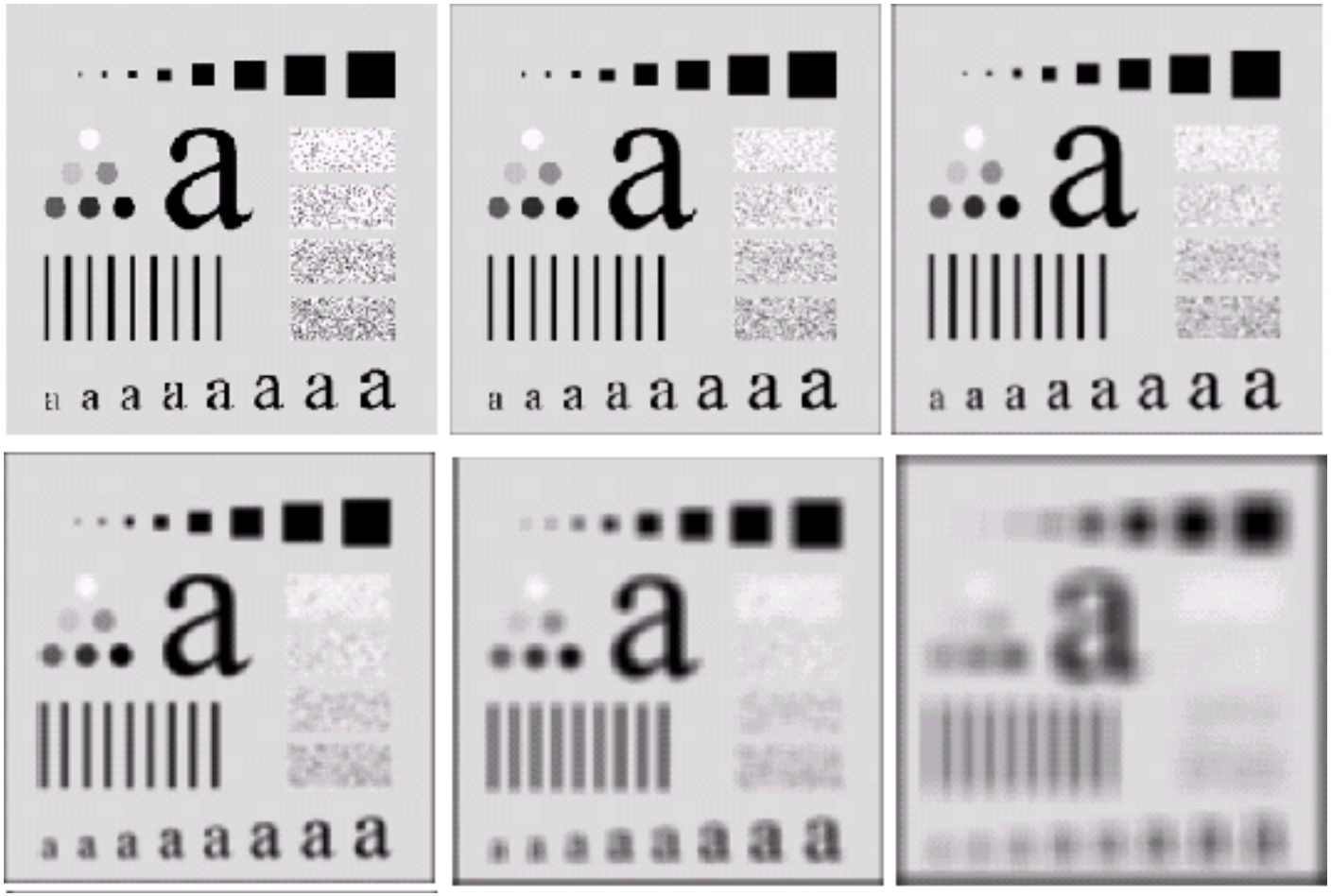
$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- All about the tradeoff: second filter preserves edges better but is not as capable of reducing noise, and the first filter is more capable of reducing noise but not good at preserving edges

Order-Statistic Filters

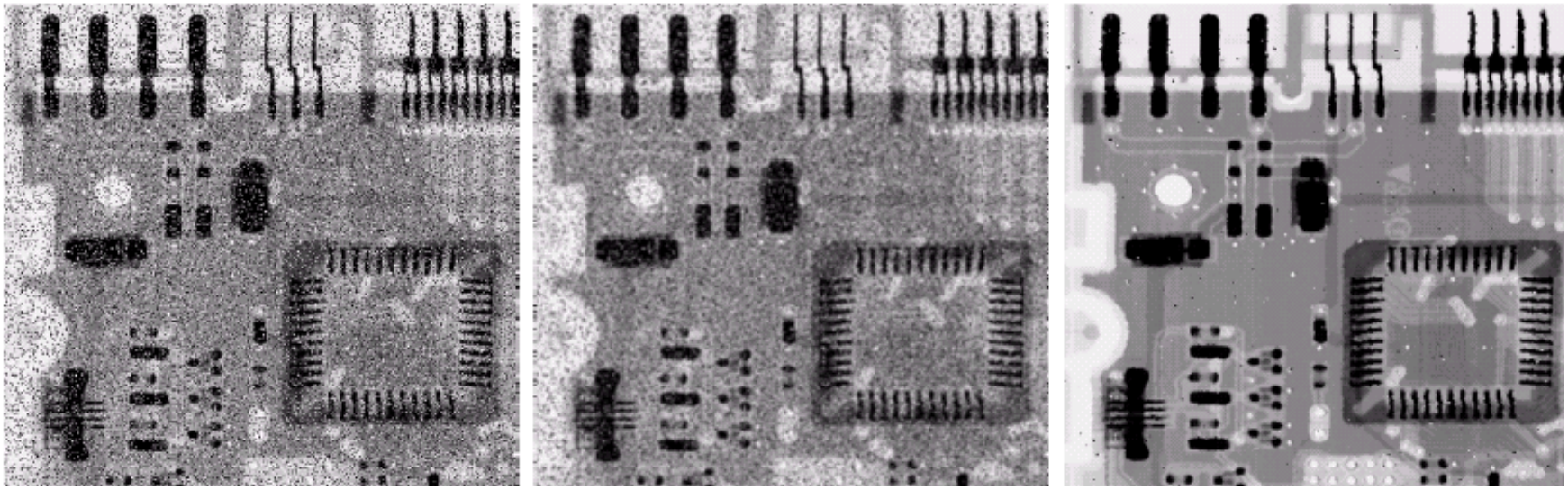
- Order-statistic filter are non-linear spatial filters based on the ordering of the pixel values
- The pixel values that are within a filter/mask range are first sorted and then a pixel among those is selected as the output of the filter
- Examples
  - Median: the pixel values are sorted and the value in the middle is selected
  - Max: the pixel values are sorted and the maximum is selected

Linear Smoothing Filters: Image Examples



Top-left: original, from top to bottom and left to right: linear smoothing with increasing filter size

Order Statistic Filters: Image Examples



Left: image corrupted by salt pepper noise, middle: linear smoothing filtering, right: median filtering

Linear Sharpening Filters

- Now, in contrast to smoothing, we want to emphasize sharp transitions
- This will allow us to enhance edges resulting in a more crisp image
- Smoothing the images $\rightarrow$ averaging
- Averaging is an approximation to integration
- Since sharpening is the inverse, it makes sense to use an approximation of derivatives for sharpening

Linear Sharpening Filters: Approximation of Derivatives

- There are several approximations to derivatives in the discrete domain
- The approximations we use must satisfy certain conditions
  - It must be zero in flat regions
  - It must be non-zero along ramps (edges and level changes)
- Second derivative approximations must be
  - zero in flat regions
  - non-zero along ramps
  - zero in regions with constant slope

Linear Sharpening Filters: Approximation of Derivatives (Cont.)

- One definition would be

$$f'(x) = f(x+1) - f(x)$$

- Applying the definition again we obtain the second derivative

$$\begin{aligned} f''(x) &= f'(x+1) - f'(x) \\ &= f(x+2) - f(x+1) - f(x+1) + f(x) \\ &= f(x+2) - 2f(x+1) + f(x) \end{aligned}$$

- Sometimes it is useful to consider the following instead (centering around the pixel location x)

$$f''(x) = f(x+1) - 2f(x) + f(x-1)$$

Linear Sharpening Filters: Comparison of Derivatives

- First order derivatives are non-zero along ramps (edges) hence result in thicker edges
- Second order derivatives are zero along ramps hence result in double and thin edges
- Second order derivatives are stronger in stressing fine details

Linear Sharpening Filter: The Laplacian

- Laplacian filters are obtained by approximating the laplacian

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

- Using the second order derivative definition we obtain the laplacian filter

$$L(x,y) = f(x+1,y) + f(x-1,y) + f(x,y-1) + f(x,y+1) - 4f(x,y)$$

Linear Sharpening Filter: The Laplacian (Cont.)

- The mask that performs this operation is

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

- We can have variations, for example including diagonal terms

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

Laplacian Combined With the Original Image

- Laplacian improves edges but destroys information in the smooth regions
- These smooth regions are present in the original image
- So if we combine the two, we obtain an image with
  - improved edges
  - smooth regions preserved
- Mathematically the enhanced image is

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

Laplacian Combined With the Original Image (Cont.)

- The mask of this operation can be obtained as

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

resulting in

$$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 5 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

Modification of the Laplacian Combined With the Original

- Sometimes it is a good idea to add the multiple of the original value of the image to improve overall brightness
- Let A denote this multplication coefficient then the mask becomes

$$\begin{bmatrix} 0 & -1 & 0 \\ 0 & A+4 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

- We can include the diagonal terms and have

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

Sharpening using unsharp masking

- Another way to obtain sharpening is to subtract a blurred version of the image from itself

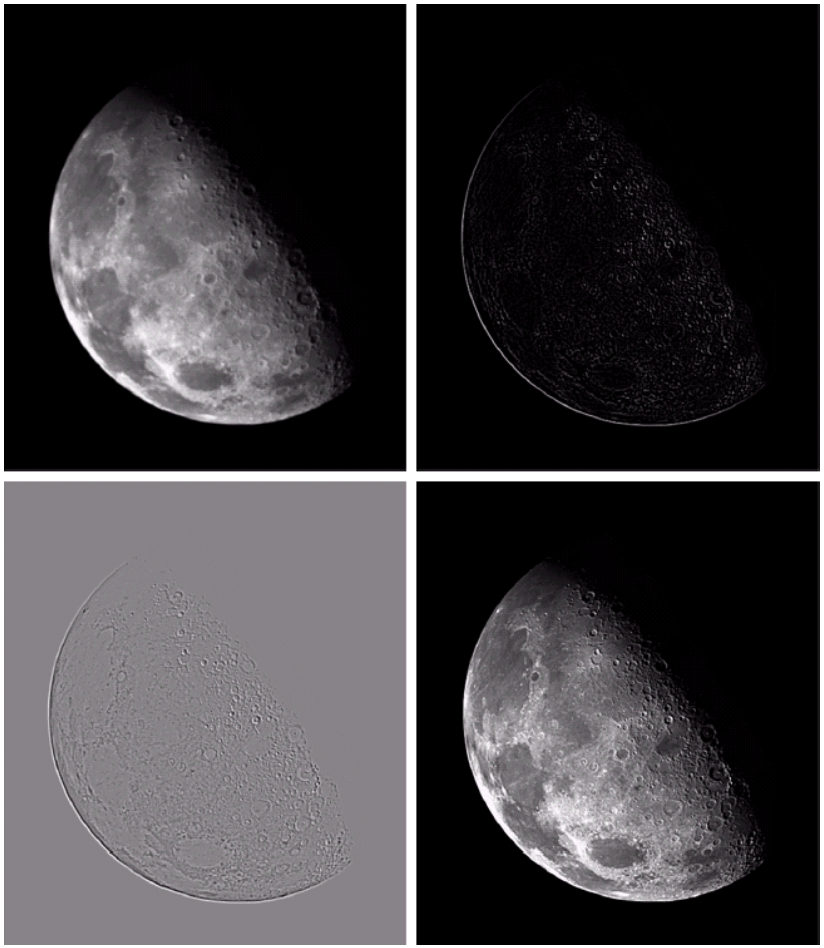
$$g(x,y) = f(x,y) - \bar{f}(x,y)$$

  where $\bar{\cdot}$ denotes blurring

- We can extend this operation by multiplying the image by a coefficient larger than one, useful to increase the overall brightness of the image

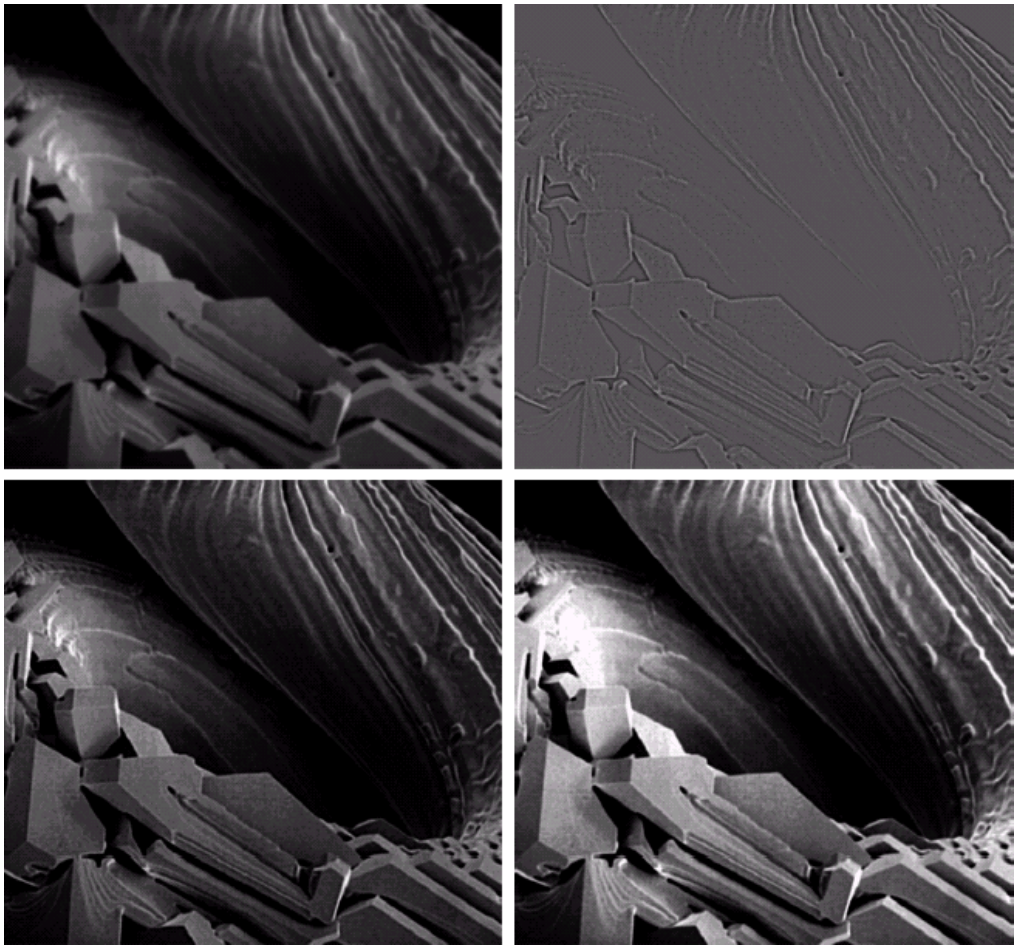
$$g(x,y) = Af(x,y) - \bar{f}(x,y)$$

Image Examples



Top-left: original, top-right: laplacian, bottom-left: scaled laplacian, bottom-right: laplacian and original together

Image Examples



Top-left: original, top-right: laplacian, bottom-left: laplacian and original, bottom-right: laplacian and original with $A > 1$, boosted

Sharpening Filters: Gradient

- We can use the first-order derivative as well for sharpening
- Gradient along x and y directions are used to create a sharpening filter
- We will use the magnitude of the gradient vector, which we call "gradient mask"
- Mathematically the output is

$$\begin{aligned} G(x,y) &= |\nabla f(x,y)| \\ &= \left[f_x^2(x,y) + f_y^2(x,y)\right]^{1/2} \end{aligned}$$

  where subscripts $_x$ and $_y$ denote partial derivatives along the x and y directions

- This operation is apparently nonlinear

Sharpening Filters: Gradient (Cont.)

- Another way to create a sharpening filter using the gradient vector is the following

$$G(x, y) = |f_x(x, y)| + |f_y(x, y)|$$

- Computationally more efficient than the magnitude of the gradient

Sharpening Filters: Gradient (Cont.)

- Notation

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

- We can use the following approximation for the derivatives

$$f_x(x,y) = f_6 - f_5 \qquad f_y(x,y) = f_8 - f_5$$

- Then the absolute gradient mask can be obtained by using the following masks

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Sharpening Filters: Gradient (Cont.)

- Another definition of the difference would be cross differences resulting in the masks

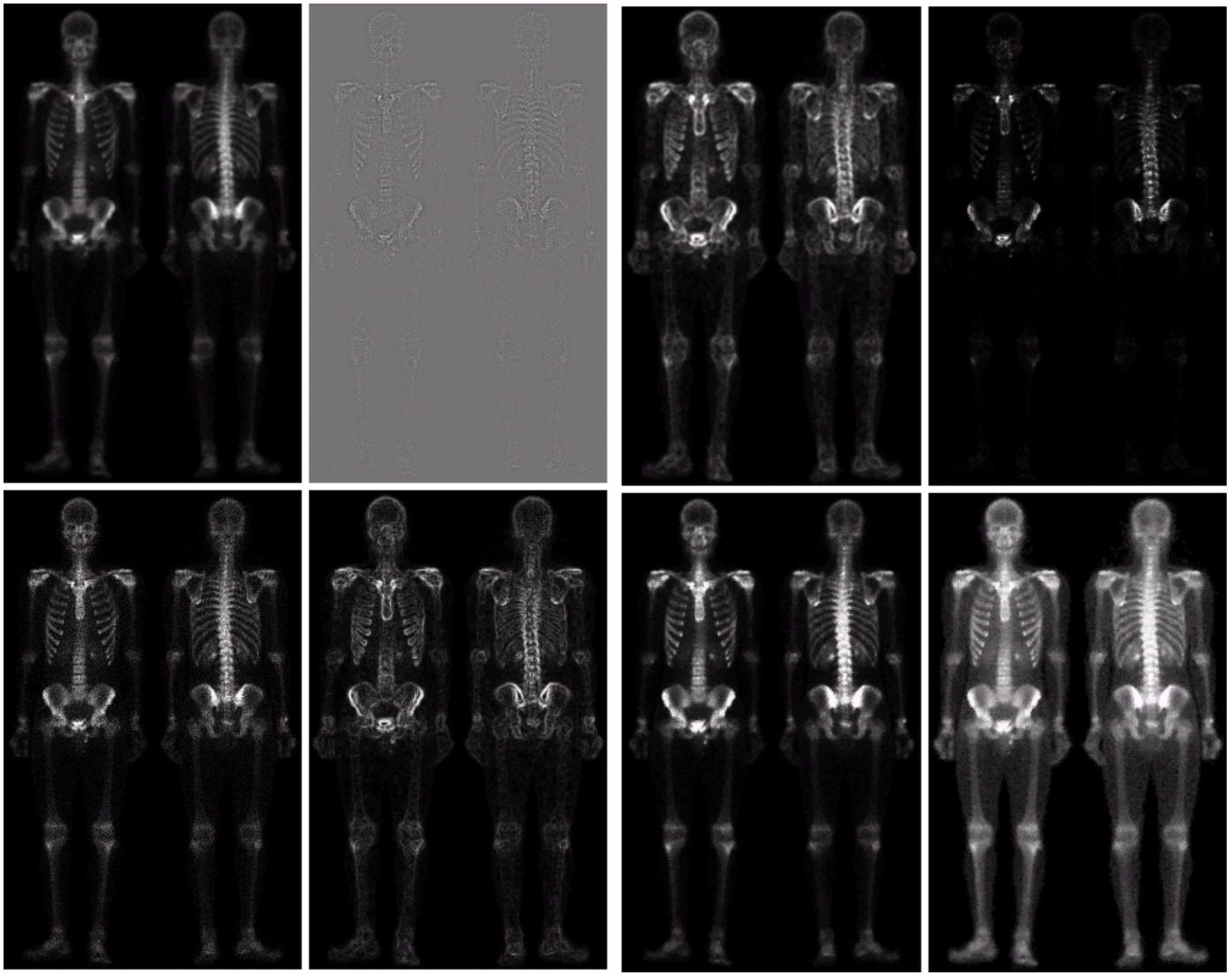
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$$

Sharpening Filters: Gradient (Cont.)

- We can extend the definition of the derivative operation to include neighboring pixels instead of just one pixel
- The resulting masks are called the "Sobel" operators

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Combination of Masks: Image Examples



See next slide.

Combination of Masks: Image Examples Explanations

Top:

- original,
- laplacian,
- sobel+smoothing,
- product of sobel+smoothing and laplacian+original

Bottom:

- laplacian+original,
- sobel,
- original + (product of sobel+smoothing and laplacian+original),
- power law transformation of original + (product of sobel+smoothing and laplacian+original),

Image Enhancement in the Frequency Domain: Introduction

- In this case, we basically first take the FT of the image and then apply our masks, finally use inverse FT to obtain the processed/enhanced image
- Remember we first multiply by $(-1)^{x+y}$ to have the FT at the center
- We can write mathematically then

$$g(x,y) = (-1)^{x+y} \times \mathcal{F}^{-1}\{H(u,v) \times \mathcal{F}\{(-1)^{x+y} f(x,y)\}\}$$

- Low-freq components $\rightarrow$ smooth regions
- High-freq components $\rightarrow$ abrupt changes, such as edges

Image Enhancement in the Frequency Domain: Relationship With Spatial Enhancement

- We know that multiplication (filtering) in the frequency domain corresponds to convolution in the spatial domain

$$\begin{aligned} g(x,y) &= f(x,y) * h(x,y) \\ &= \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) h(x-m, y-n) \end{aligned}$$

- Notice that this is similar to filtering in the spatial domain with small changes
- Then filtering in the spatial domain and filtering in the frequency domain are equivalent provided that the FT pairs are used

$$h(x,y) * f(x,y) \Leftrightarrow H(u,v) \times F(u,v)$$

- Filtering in frequency domain is more intiutive, and filterin in spatial domain is generally faster
- Design in the frequency domain and implement in spatial domain

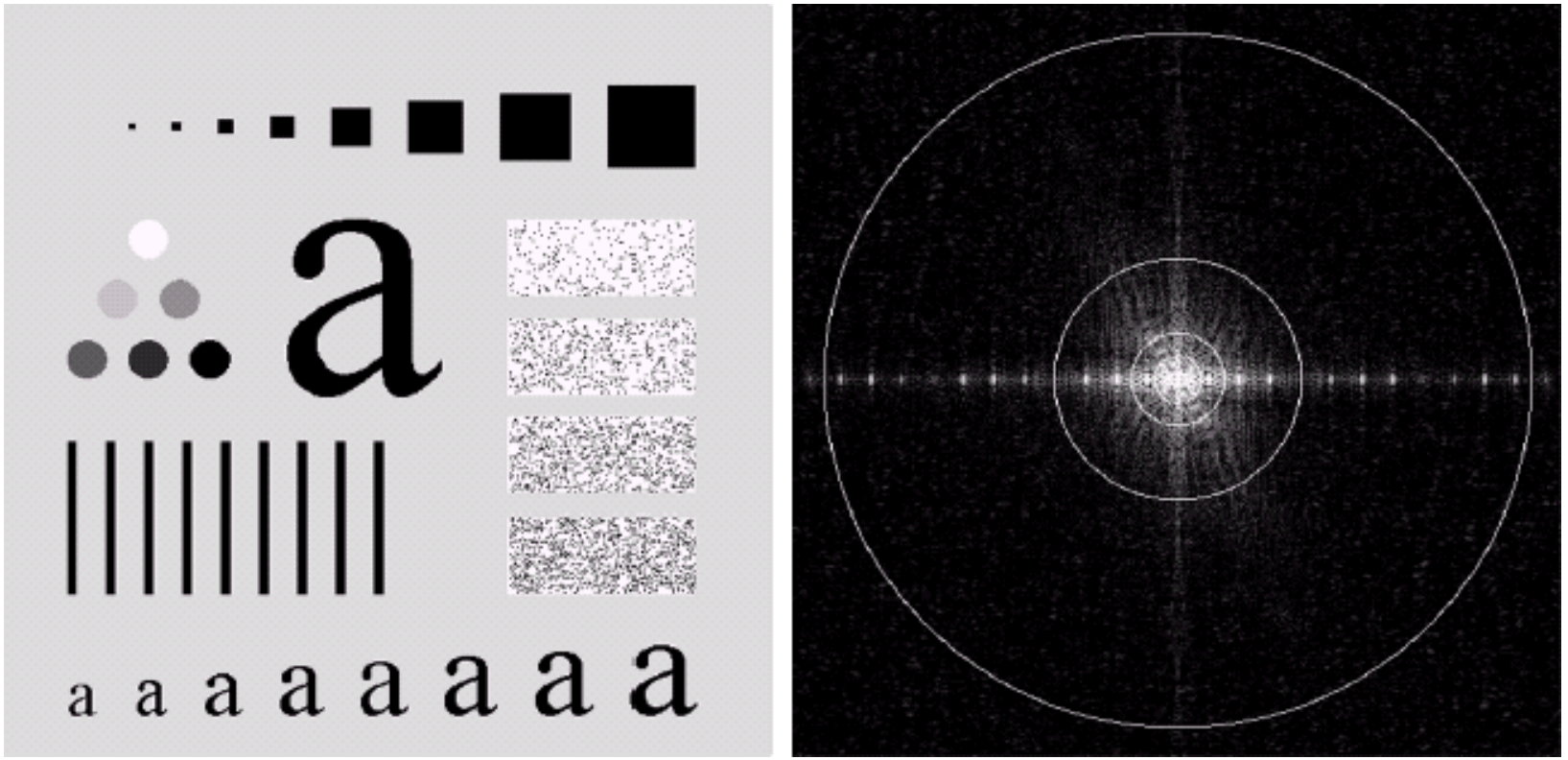
Smoothing in the Frequency Domain

- Let us consider a few examples of low-pass filter
- Ideal low pass filter: It has a ideal circular pass region

$$H(u,v) = \begin{cases} 1 & D(u,v) \leq D_0 \\ 0 & D(u,v) > D_0 \end{cases}$$

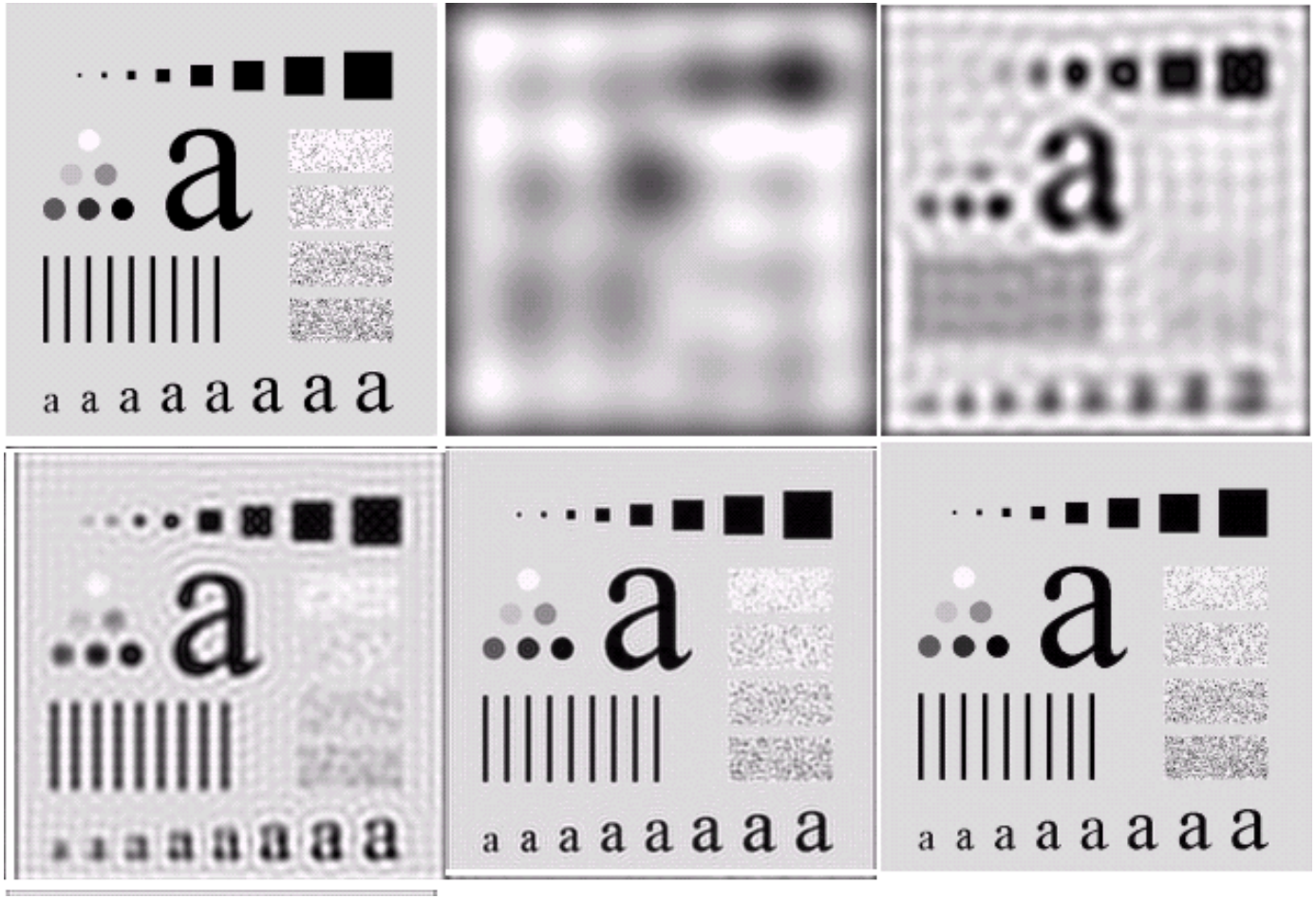
  where $D(u,v)$ is the euclidian distance to the center, and D_0 is the parameter that determines the passing frequencies
- Radially symmetric: it is sufficient to specify the filter along a line passing through center
- One way to characterize filters is by determining how much power do the filters pass

Smoothing in the Frequency Domain (Cont.)



Left: image, right: its FT, circles showing enclosed powers 99.5, 98, 96.4, 94.6, 92.0 percentage

Smoothing in the Frequency Domain: Ideal Low Pass Filtering Image Example

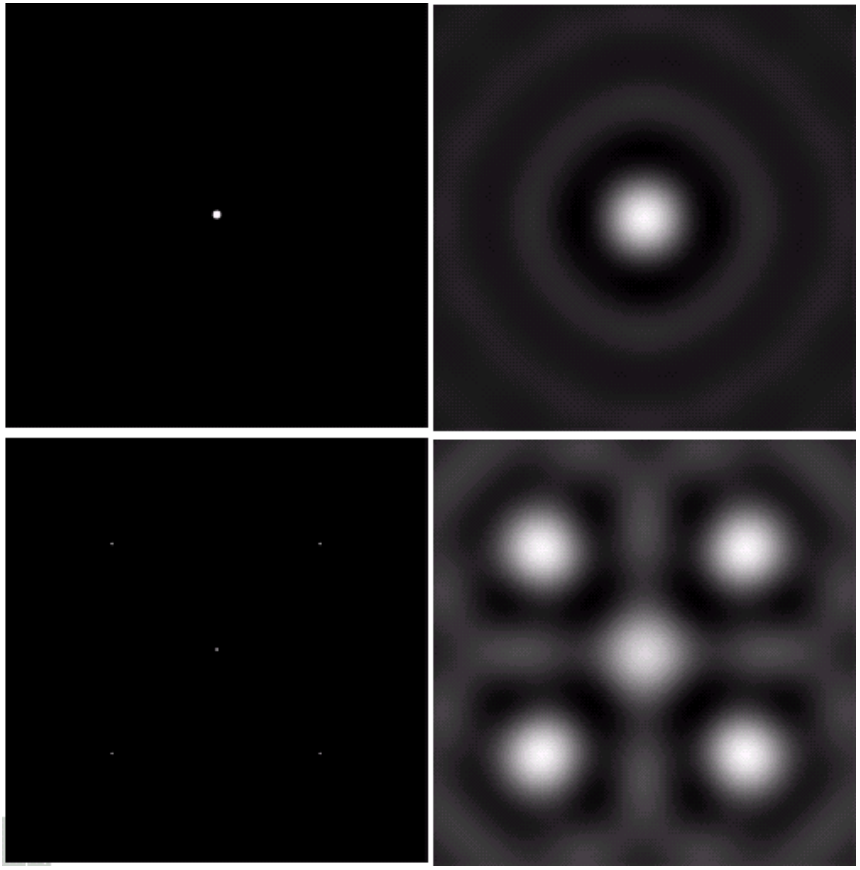


Low pass filtering with different cutoff frequencies. Top-left original, others filtered with 8, 5.4, 3.6, 2.0, 0.5 percentage removed

Smoothing in the Frequency Domain: Relationship to Spatial Filtering

- We can obtain the spatial version of the ideal low pass filtering by
  - multplying by $(-1)^{u+v}$
  - taking the inverse FT
  - multiplying by $(-1)^{x+y}$

Smoothing in the Frequency Domain: Ideal Low Pass Filtering in the Spatial Domain



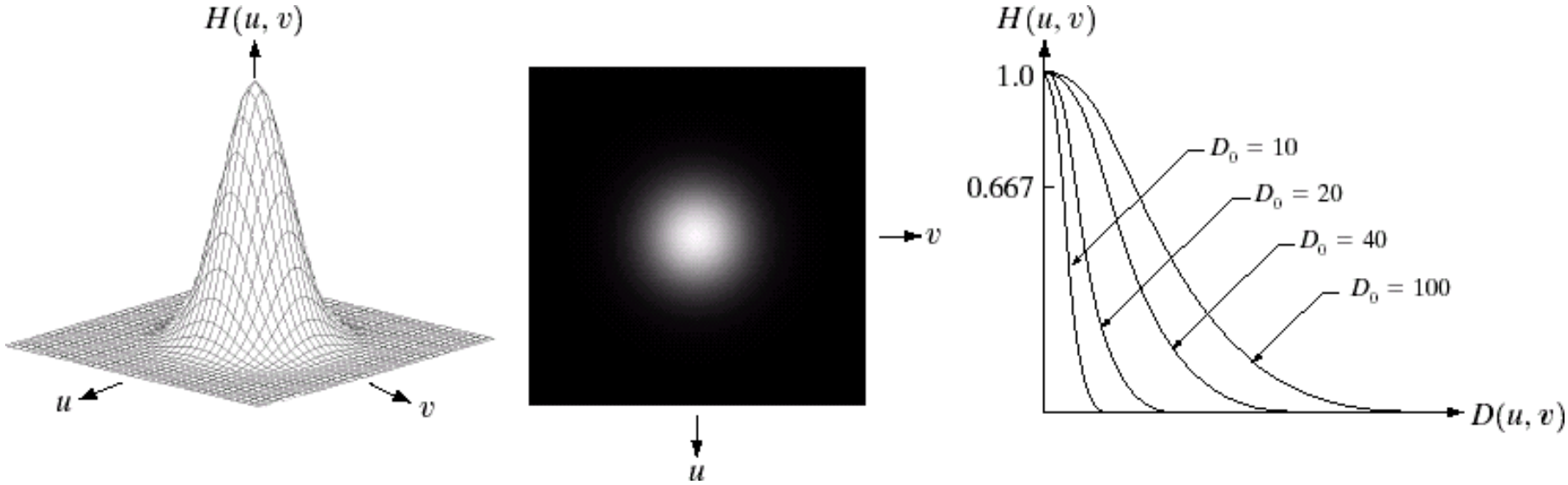
Top-left: ideal low pass filter in the frequency domain, top right: ideal low pass filter in spatial domain, bottom-left: test image, 5 impulses, bottom-right: filtered version of the test image

Smoothing in the Frequency Domain: Gaussian Filter

- Gaussian low pass filter which has a much more smoother functional form is

$$H(u,v) = e^{-D^2(u,v)/2\sigma^2}$$

- We can select $\sigma = D_0$
- Inverse FT of a Gaussian is another Gaussian, therefore the shape of the filter in the spatial and the frequency domain are similar



Smoothing in the Frequency Domain: Butterworth Filter

- Ideal LP filter has a very sharp functional form and the Gaussian filter has a very smooth functional form
- Butterworth is somewhere in between

$$H(u,v) = \frac{1}{1 + [D(u,v)/D_0]^{2n}]}$$

where n determines the order also affecting the sharpness/smoothness