Intensity Transformations

--Content not available till here--

Lowpass Gaussian Filter Kernels

- Because of their simplicity, box filters are suitable for quick experimentation and they often yield smoothing results that are visually acceptable. They are useful also when it is desired to reduce the effect of smoothing on edges.
- However, box filters have limitations that make them a poor choice, for example, a defocused lens is often modeled as a lowpass filter, but box filters are poor aproximations to the blurring characteristics of lenses.
- Another limitation is the fact that box filters favor blurring along perpendicular directions. In applications involving images with high level of detail, or with strong geometric components, the directionality of box filters often produces undesirable results.

As it turns out, gaussian kernels of the form

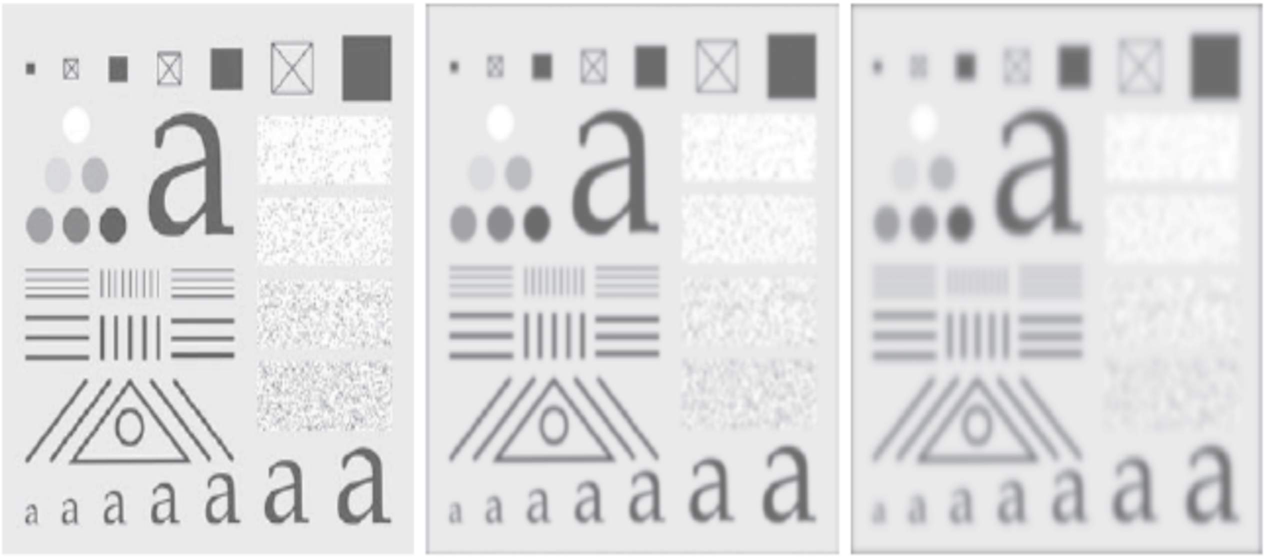
$$w(s,t) = G(s,t) = Ke^{-\frac{s^2+t^2}{2\sigma^2}}$$

are the only circularly symmetric kernels that are also seperable. Thus, because Gaussian kernels of this form are seperable, they enjoy same computational advantages as box kernels, but also have a host of additional properties.

By letting $r = [s^2 + t^2]^{1/2}$ or $r = \sqrt{s^2 + t^2}$, we can write

$$G(r) = Ke^{-\frac{r^2}{2\sigma^2}}$$

Variable r is the distance from the center to any point on the function G (basically r = radius). Because we work generally with odd kernel sizes, the center of such kernels fall on integer values , and it follows that all values of r^2 are integers too.



a b c

FIGURE 3.36 (a)A test pattern of size 1024×1024. (b) Result of lowpass filtering the pattern with a Gaussian kernel of size 21×21, with standard deviations $\sigma = 3.5$. (c) Result of using a kernel of size 43×43, with $\sigma = 7$. This result is comparable to Fig. 3.33(d). We used $K = 1$ in all cases.

Values of a gaussian function at a distance larger than 3σ from the mean are small enough that they can be ignored. So, if we select the size of a Gaussian kernel to be $\lceil 6\sigma \rceil \times \lceil 6\sigma \rceil$, we are assured of getting essentially the same result if we had used an arbitrary large Gaussian kernel. We would use the smallest *odd* integer that satisfies this condition (a 43×43 kernel, if $\sigma = 7$)

2 other fundamental properties of Gausssian functions are

- Product of 2 Gaussian functions is a Gaussian function
- Convolution of 2 Gaussian functions is a Gaussian function

Gaussian kernels have zero mean, so our interest is in standard deviation
$* = convolution, \times = product$

	f	g	$f \times g$	$f * g$
Mean	m_f	m_g	$m_{f\times g} = \frac{m_f\sigma_g^2 + m_g\sigma_f^2}{\sigma_f^2+\sigma_g^2}$	$m_{f*g} = m_f + m_g$
Standard Deviation	σ_f	σ_g	$\sigma_{f\times g} = \sqrt{\frac{\sigma_f^2\sigma_g^2}{\sigma_f^2+\sigma_g^2}}$	$\sigma_{f*g} = \sqrt{\sigma_f^2 + \sigma_g^2}$

Filtering is sometimes performed successive stages, and the same result can be obtained by one stage of filtering with a composite kernel formed as the convolution of the individual kernels ($w = w_1 * w_2 * w_3 * .... * w_n$, we need to calculate w first and then SD and mean, but with the above we can do it very easily without convolution first). If the kernels are gaussian, we can use the above result to compute standard deviation of the composite kernel (and thus completely define it, since SD and mean completely defines it) without actually having to perform the convolution of all the individual kernels.

a b c

FIGURE 3.37 (a) Result of filtering Fig. 3.36(a) using a Gaussian kernels of size 43×43, with $\sigma = 7$. (b) Result of using a kernel of 85×85, with the same value of σ. (c) Difference image.

Mirror and Replicate Padding

Mirror Padding

- Also called **symmetric padding**.
- Values outside the boundary of the image are obtained by mirror-reflecting the image across its border.
- Useful when when the areas near the border contain image details.

Replicate Padding

- Values outside the boundary are set equal to nearest image border value.
- Useful when the areas near the border of the image are constant.

Lowpass filtering is a rugged, simple method for estimating shading patterns. Gaussian filtering is a type of lowpass filtering.

Image A = original image
Image B → Lowpass Filter → Shading estimation
Image A / shading estimation = Shaded corrected image

Order-statistic (Nonlinear) filters

- Non linear
- Based on ordering (ranking) the pixels contained in the filter mask.
- Replacing the value of the center pixel with the value determined by the ranking filter.
- Example: Median filter, min filter, max filter.

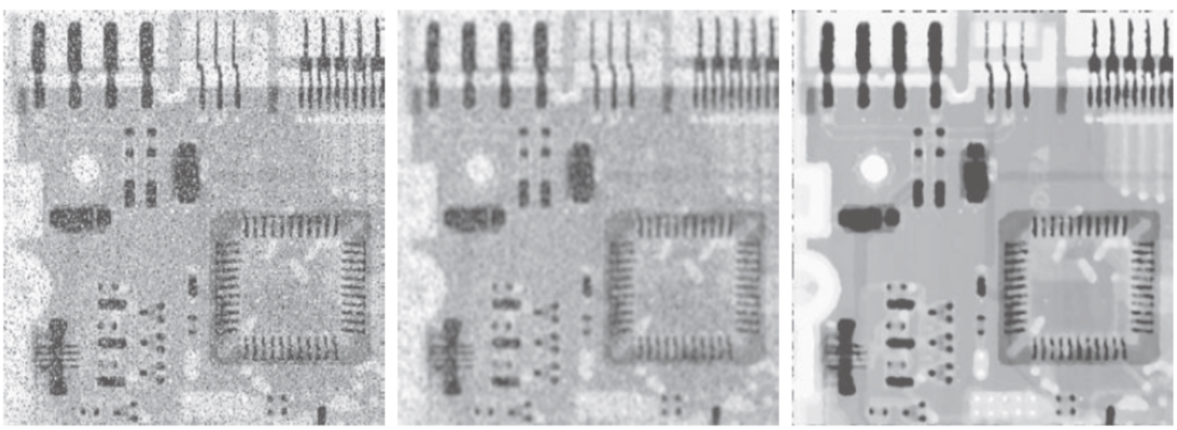
Median Filter

As its name applies, it replaces the value of the center pixel by the median of intensity values in the neighborhood of that pixel (including the center value).

- Provides excellent noise reduction capabilities for certain types of random noise, with considerably less blurring than linear smoothing filters of similar size.
- Particulary effective in the presence of impulse noise (a.k.a. salt-and-pepper noise, when it manifests itself as black and white dots superimposed on an image).

Calculation

- The median, ξ, of a set of values is such that half the values in the set are less than equal to ξ and half the values are greater than or equal to ξ.
- First, we sort the values of the pixels in the neighborhood
- Next we determine the median and assign that value to the pixel in the filtered image corresponding to the location of center of the neighborhood.
- For example: Suppose a 3×3 neighborhood, $values = [10, 20, 20, 15, 20, 20, 25, 100]$
  On Sorting, $values = [10, 15, 20, 20, 20, 20, 25, 100] \rightarrow median = 20$
- Thus, the principle function of the median filter is to force points to more like their neighbors.



a b c

FIGURE 3.43 (a) X-ray image of a circuit board, corrupted by salt-and-pepper noise. (b) Noise reduction using a 19×19 Gaussian lowpass filter kernel with $\sigma = 3$. (c) Noise reduction using a 7×7 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Sharpening (highpass) Spatial filters

- Foundation
- Laplacian operator
- Unsharp masking and highboost filtering
- Using first-order derivatives for nonlinear image sharpening

Foundation

First order derivative,

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

Second order derivative,

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$

Zero Crossing

A **zero-crossing** is a point where the sign of the second derivative changes (e.g. from positive to negative), represented by an intercept of the axis (zero value) in the graph of the function. This is quite useful in detecting edges.

Laplacian Operator

Simplest isotropic derivative operator is the laplacian. For 2 variables x and y, it is defined as:

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

Derivatives are linear $\Rightarrow$ Laplacian is linear

In the discrete form, in x-direction:

$$\frac{\partial^2 f}{\partial x^2} = f(x+1,\ y) + f(x-1,\ y) - 2f(x,\ y)$$

Similary, in the y-direction:

$$\frac{\partial^2 f}{\partial y^2} = f(x,\ y+1) + f(x,\ y-1) - 2f(x,\ y)$$

Adding these two,

$$\nabla^2 f = f(x+1,\ y) + f(x-1,\ y) + f(x,\ y+1) + f(x,\ y-1) - 4f(x,\ y)$$

Kernels used in Laplacian

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

The 1^{st} one is used implement the equation above. Yields the same results when image is rotated in increments of 90°.
The 2^{nd} one is used to implement an extension of the above equation that includes the diagonal terms. Yields the same results when image is rotated in increments of 45°.
Last two with $c = 1$ described shortly.

The diagonal directions can be incorporated in the definition of the digital Laplacian by adding four more terms to the equation above. Because each diagonal term would contains a $-2f(x, y)$ term, the total subtracted from the difference terms now would be $-8f(x, y)$.

Since laplacian is derivative operator, it highlights sharp intensity transitions (edges)in an image and de-emphasizes regions of slow varying intensities. This will tend to produce images with grayish edge lines and other discontinuities, all superimposed on dark, featureless background.

So, the output image will be:

$$g(x,\ y) = f(x,\ y) + c[\ \nabla^2 f(x,\ y)\ ]$$

Unsharp Masking and Highboost Filtering

Unsharp Masking

Sharpen images consists of subtracting an unsharp (smoothed) version of an image from the original image.

STEPS:

- Blur the original image.
- Subtract the blurred image from the original
- Add mask to the original

$\bar{f}(x,\ y) =$ blurred image, then

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

Then, add a weighted portion to the original,

$$g(x,\ y) = f(x,\ y) + k \times g_{mask}(x,\ y),\ k \geq 0$$

$k > 1 \Rightarrow$ **Highboost filtering**

a b c
d e

FIGURE 3.49 (a) Original image of size 600×259 pixels. (b) Image blurred using a 31×31 Gaussian lowpass filter with $\sigma = 5$. (c) Mask. (d) Result of unsharp masking using Eq. (3-56) with $k = 1$. (e) Result of highboost filtering with $k = 4.5$.

Using the first-order derivative for sharpening (gradient)

For $f(x,\ y)$, the gradient (gradient is a vector) is defined as

$$\nabla f \equiv grad(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

The magnitude $M(x,\ y)$ is

$$M(x,\ y) = mag(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

$$M(x,\ y) \approx |\ g_x\ | + |\ g_y\ |$$

For a matrix,

$$\begin{bmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{bmatrix} \quad M(x,\ y) \approx |\ z_8 - z_5\ | + |\ z_6 - z_5\ |$$

Roberts Cross-gradient operators

$$M(x,\ y) \approx |\ z_9 - z_5\ | + |\ z_8 - z_6\ |$$

Sobel Operators

$$M(x,\ y) \approx |\ (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)\ | + |\ (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7$$

Roberts cross and Sobel operators as kernels

$$\begin{bmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{bmatrix}$$

KERNELS FOR ROBERTS-CROSS

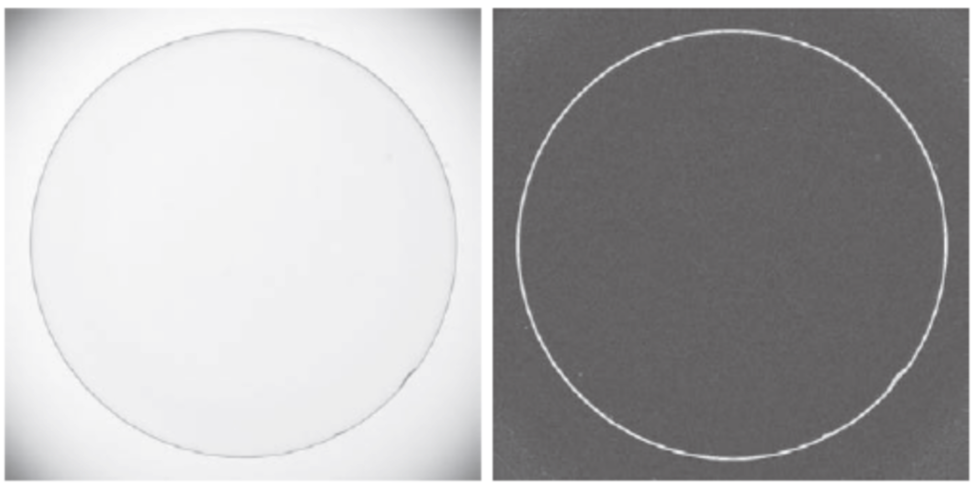
$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \quad (z_5 \times -1) + (z_6 \times 0) + (z_8 \times 0) + (z_9 \times 1) = -z_5 + z_9 \equiv z_9 -$$

KERNELS FOR SOBEL

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

a b

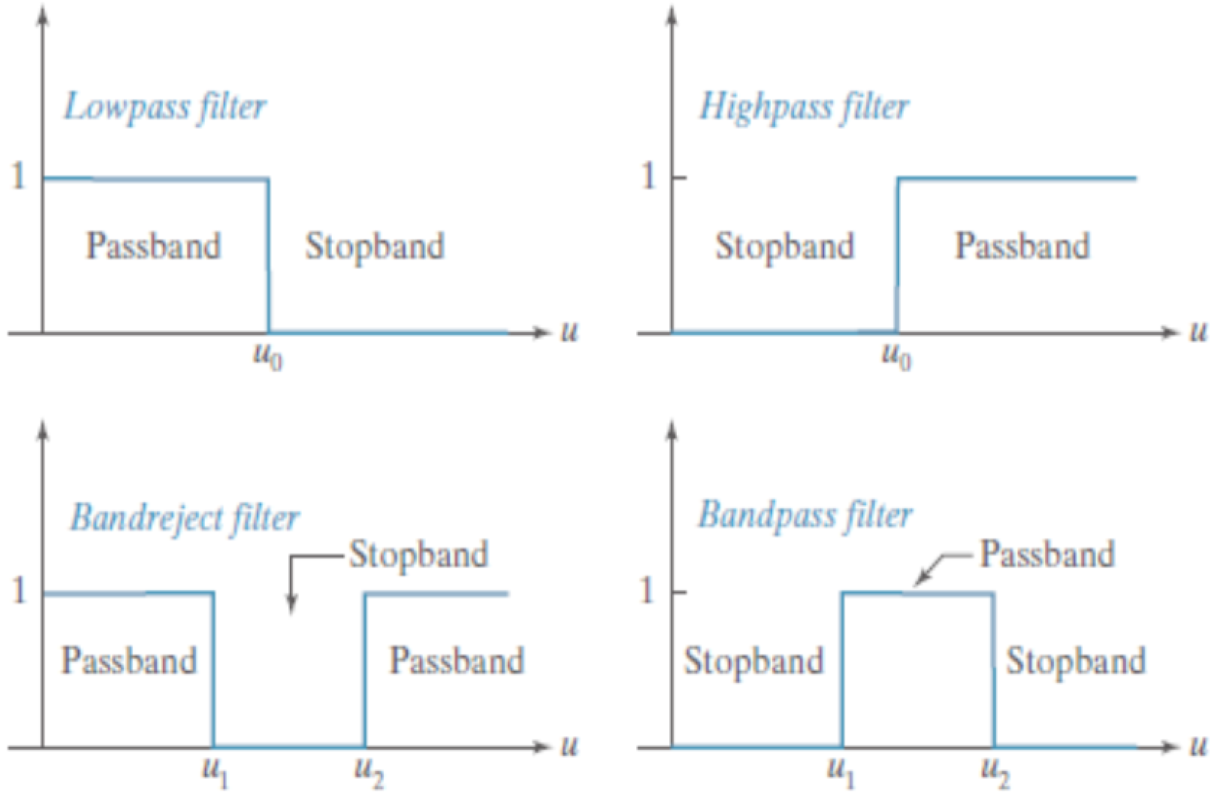
FIGURE 3.51
(a) Image of a contact lens (note defects on the boundary at 4 and 5 o'clock).
(b) Sobel gradient.
(Original image courtesy of Perceptics Corporation.)



Highpass, Bandreject, Bandpass filters from Lowpass filter

Spatial and frequency domain are classified into 4 broad categories:

- **Lowpass** and **Highpass** Filters
  - Lowpass filters filters out high frequencies
  - Highpass does the opposite, filters out all frequencies below a cut-off value u_0.
- **Bandpass** and **Bandreject** Filters
  - A band-pass filter or bandpass filter is a device that passes frequencies within a certain range and rejects frequencies outside that range.
  - Band-reject filters (also called band-stop filters) passes frequencies that lie outside a certain range and reject those which lie in that range.



For Highpass

- Calculate lowpass at u_0.
- Subtract the above from the whole image.

For Bandreject

- Calculate lowpass at u_1.
- Calculate highpass at u_2 using lowpass.
- Add the above results.

For Bandpass

- Calculate lowpass at u_1.
- Calculate highpass at u_2 using lowpass.
- Subtract both of the above results from the whole image.

Summary

Filter Type	Spatial Kernel in terms of lowpass kernel, lp
Lowpass	$lp(x,\ y)$
Highpass	$hp(x,\ y) = \delta(x,\ y) - lp(x,\ y)$
Bandreject	$br(x,\ y) = lp_1(x,\ y) + hp_2(x,\ y)$ $= lp_1(x,\ y) + [\delta(x,\ y) - lp_2(x,\ y)]$
Bandpass	$bp(x,\ y) = \delta(x,\ y) - br(x,\ y)$ $= \delta(x,\ y) - [lp_1(x,\ y) + [\delta(x,\ y) - lp_2(x,\ y)]]$