Spatial filtering

- Use of spatial masks for filtering is called **spatial filtering**
  - May be linear or nonlinear
- 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 filters (examples)

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

Spatial filter masks

- If the center of the mask is at \((x,y)\), then the pixel value at \((x,y)\) is replaced by \(R\)
- This continues until all pixels in the image are covered
- For pixels near the boundary of the image, \(R\) may be computed using partial neighborhoods or by padding the input appropriately
- Usual practice is to create a new image with the values of \(R\) (as opposed to modifying pixels in the original image)

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
    \[ R = \max \{z_k \mid k = 1,2,\ldots,9\} \]
  - **Min filter:** Used to find the dimmest points in an image
    \[ R = \min \{z_k \mid k = 1,2,\ldots,9\} \]

Smoothing 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

\[
\begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 
\end{bmatrix}
\]
Averaging filter example

Smoothing filter examples

Smoothing filters (continued)

• 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.

Median filtering example

Sharpening filters

• The shape of the impulse response needed to implement a highpass (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.

Sharpening filters (continued)

• 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 somewhat 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 \([0, L-1]\) (or \([1,256]\) for a 256-gray-level colormap MATLAB image)
Highpass filter example

- A highpass filter may be computed as:
  \[ \text{Highpass} = \text{Original} - \text{Lowpass} \]
- Multiplying the original by an amplification factor yields a highboost or high-frequency-emphasis filter
  \[ \text{Highboost} = (A - 1) \text{Original} + \text{Original} - \text{Lowpass} \]
  \[ = (A - 1) \text{Original} + \text{Highpass} \]
  - If \( A > 1 \), part of the original image is added to the highpass 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 \( w = 9A - 1 \) (\( A \geq 1 \))

Sharpening filters (continued)

Highboost filter example

- A basic first-order derivative of a 1-D function, \( f(x) \), is the difference
  \[ \frac{df}{dx} = f(x + 1) - f(x) \]
- The second-order derivative of \( f(x) \) is the difference
  \[ \frac{d^2f}{dx^2} = f(x + 1) + f(x - 1) - 2f(x) \]

Derivative Functions

- Differentiation may be used for edge enhancement (detail sharpening)
- The most common method for differentiating is the gradient.
- For \( f(x,y) \) the gradient of \( f \) at \( (x,y) \) is defined as:
  \[ \nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right] \]
- With a magnitude of:
  \[ |\nabla f| = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2} \]
Derivative filters (continued)

The magnitude of the gradient at $z_5$ can be approximated in a number of ways:

- The simplest is to use the difference $(z_5 - z_8)$ in the x direction and $(z_5 - z_6)$ in the y direction.
- Or approximated as:
  \[ \frac{z_1 - z_2 + z_3 - z_4 + z_5 - z_6 + z_7 - z_8}{2} \]

Derivative filters (continued)

- A cross difference may also be used to approximate the magnitude of the gradient:
  \[ \sqrt{(z_5 - z_8)^2 + (z_5 - z_6)^2} \]
- This can be implemented by taking the absolute value of the response of the following two masks (the Roberts cross-gradient operators) and summing the results:
  \[ \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \]

Derivative filters (continued)

- Extension to a 3x3 mask yields the following:
  \[ \nabla f = \left| (z_1 + z_2 + z_3) - (z_5 + z_6 + z_7) \right| + \left| (z_1 + z_2 + z_3) - (z_5 + z_6 + z_7) \right| \]
- The difference between the first and third rows approximates the derivative in the x direction.
- The difference between the first and third columns approximates the derivative in the y direction.
- The Prewitt operator masks may be used to implement the above approximation:
  \[ \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} \]

Derivative filters (continued)

- The Sobel operator masks may also be used to implement the derivative approximation:
  \[ \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \]
- The Sobel operators are widely used for edge detection (will be discussed in more detail later).
- NOTE: All the mask coefficients for all the derivative filters sum to zero, indicating a 0 response in a constant area (as expected of a derivative operator).

Derivative filters (continued)

- After application of a derivative filter, generally the output will be scaled (as in the lowpass and highpass cases).
- The result is then thresholded by setting any values above the threshold to white (256 in MATLAB with a 256 gray-level colormap) and all below the threshold are set to black (1 in MATLAB) or left with their initial values in $f(x,y)$.
- The derivative filters may be applied:
  - Only in the x direction
  - Only in the y direction
  - In both directions (taking either the sum or maximum of the responses from the filter masks).

Derivative filter example

- Original image
- Sobel filtered image:
  \[ g = \text{derivativefilter}(f, 'Sobel', 25, 'sum', 0); \]
Derivative filter example (continued)

- Sobel filtered image
  \( g = \text{derivativefilter}(f, \text{"Sobel"}, 15, \text{"horiz"}, 0); \)
- Sobel filtered image
  \( g = \text{derivativefilter}(f, \text{"Sobel"}, 10, \text{"vert"}, 0); \)

Derivative filter example (continued)

- Sobel filtered image
  \( g = \text{derivativefilter}(f, \text{"Sobel"}, 15, \text{"horiz"}, 0); \)
- Sobel filtered image
  \( g = \text{derivativefilter}(f, \text{"Sobel"}, 10, \text{"max"}, 0); \)

- Sobel filtered image
  \( g = \text{derivativefilter}(f, \text{"Sobel"}, 10, \text{"max"}, 1); \)