Image Enhancement II: Neighborhood Operations

Image Enhancement: Spatial Filtering Operation

• Idea: Use a “mask” to alter pixel values according to local operation
• Aim: (De)-Emphasize some spatial frequencies in the image.

Figure 3.37: (a) X-ray image of circuit board corrupted by salt-and-pepper noise, (b) Noise reduction with a $3 \times 3$ averaging mask, (c) Noise reduction with a $3 \times 3$ median filter. (Original image courtesy of Mr. Joseph E. Pascute, Lidi Inc.)
Overview of Spatial Filtering

• Local linear operations on an image

\[ g(x, y) = w_1 f(x-1, y-1) + w_2 f(x-1, y) + \cdots + w_8 f(x+1, y-1) + w_9 f(x+1, y+1) \]

Moving average

• We replace each pixel with a *weighted* average of its neighborhood
• The weights are called the *filter kernel*
• What are the weights for the average of a 3x3 neighborhood?

\[
\begin{array}{ccc}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array}
\]

"box filter"

Source: D. Lowe
Spatial Filtering: Blurring

**Example**

Averaging Mask:

\[
\begin{array}{ccc}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array}
\]

\[
\frac{1}{9}
\]

---

Image Enhancement: Spatial Filtering Operation

**Local linear operations on an image**

\[
g(x, y) = w_1 f(x - 1, y - 1) + w_2 f(x - 1, y) + \cdots + w_k f(x + 1, y - 1) + w_k f(x + 1, y + 1)
\]

**Input**: \( f(x,y) \), **Output**: \( g(x,y) \):

**Input Image**

**Mask**

**Output Image**

*Usually odd*
Defining convolution

- Let $f$ be the image and $g$ be the kernel. The output of convolving $f$ with $g$ is denoted $f * g$.

$$(f * g)[m, n] = \sum_{k,l} f[m-k, n-l] g[k,l]$$

- Convention: kernel is “flipped”
- MATLAB: conv2 vs. filter2 (also imfilter)

Key properties

- **Linearity**: $\text{filter}(f_1 + f_2) = \text{filter}(f_1) + \text{filter}(f_2)$

- **Shift invariance**: same behavior regardless of pixel location: $\text{filter}($shift($f$)) = shift($\text{filter}(f)$)

- Theoretical result: any linear shift-invariant operator can be represented as a convolution
Important details

- What is the size of the output?
- MATLAB: filter2(g, f, shape)
  - \textit{shape} = ‘full’: output size is sum of sizes of \( f \) and \( g \)
  - \textit{shape} = ‘same’: output size is same as \( f \)
  - \textit{shape} = ‘valid’: output size is difference of sizes of \( f \) and \( g \)

\begin{center}
\begin{tabular}{ccc}
\includegraphics[width=0.3\textwidth]{full} & \includegraphics[width=0.3\textwidth]{same} & \includegraphics[width=0.3\textwidth]{valid}
\end{tabular}
\end{center}

Image Enhancement: Spatial Filtering Operation

- An important point: \textit{Edge Effects}
  - To compute all pixel values in the output image, we need to fill in a “border”

\begin{center}
\begin{tabular}{c}
\includegraphics[width=0.4\textwidth]{mask}
\end{tabular}
\end{center}

\begin{center}
\begin{tabular}{c}
\includegraphics[width=0.4\textwidth]{border}
\end{tabular}
\end{center}

Mask dimension = 2\( M \)+1

Border dimension = \( M \)
Image Enhancement: Spatial Filtering Operation

• An important point: **Edge Effects (Ex.: 5x5 Mask)**
  – How to fill in a “border”
    • Zeros (Ringing)
    • Replication (Better)
    • Reflection (“Best”)

![Image showing different methods for filling the border](image)

• Procedure:
  – Replicate row-wise
  – Replicate column-wise
  – Apply filtering
  – Remove borders

Implementation

• What about near the edge?
  – the filter window falls off the edge of the image
  – need to extrapolate
  – methods (MATLAB):
    • clip filter (black): `imfilter(f, g, 0)`
    • wrap around: `imfilter(f, g, 'circular')`
    • copy edge: `imfilter(f, g, 'replicate')`
    • reflect across edge: `imfilter(f, g, 'symmetric')`

Source: S. Marschner
• What about near the edge?
  – the filter window falls off the edge of the image
  – need to extrapolate
  – methods:
    • clip filter (black)
    • wrap around
    • copy edge
    • reflect across edge

Source: S. Marschner
Examples of linear filters

Original

?

Source: D. Lowe

Practice with linear filters

Original

Filtered (no change)

Source: D. Lowe
Practice with linear filters

Original

![Original Image](image1)

![Filter Kernel](image2)

Shifted left
By 1 pixel

Source: D. Lowe
Practice with linear filters

Original

Blur (with a box filter)

Source: D. Lowe
Practice with linear filters

Original

\[ \begin{bmatrix}
0 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 0
\end{bmatrix} \quad - \quad \frac{1}{9} \quad \begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1
\end{bmatrix} \quad ? \]

(Note that filter sums to 1)

Source: D. Lowe

---

Sharpening filter
- Accentuates differences with local average

Original

\[ \begin{bmatrix}
0 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 0
\end{bmatrix} \quad - \quad \frac{1}{9} \quad \begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1
\end{bmatrix} \]

Source: D. Lowe
Sharpening

Source: D. Lowe

Examples of some other smoothing or "low-pass" filters:
Gaussian Kernel

\[ G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \]

- Constant factor at front makes volume sum to 1 (can be ignored, as we should re-normalize weights to sum to 1 in any case)

Source: C. Rasmussen

Choosing kernel width

- Gaussian filters have infinite support, but discrete filters use finite kernels

Source: K. Grauman
Choosing kernel width

• Rule of thumb: set filter half-width to about $3 \sigma$

Example: Smoothing with a Gaussian
Mean vs. Gaussian filtering

Gaussian filters

- Remove “high-frequency” components from the image (low-pass filter)
- Convolution with self is another Gaussian
  - So can smooth with small-width kernel, repeat, and get same result as larger-width kernel would have
  - Convolving two times with Gaussian kernel of width $\sigma$ is same as convolving once with kernel of width $\sigma\sqrt{2}$
- *Separable* kernel
  - Factors into product of two 1D Gaussians
Separability of the Gaussian filter

\[ G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \]

\[ = \left( \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x^2}{2\sigma^2}\right) \right) \left( \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{y^2}{2\sigma^2}\right) \right) \]

The 2D Gaussian can be expressed as the product of two functions, one a function of \(x\) and the other a function of \(y\).

In this case, the two functions are the (identical) 1D Gaussian.

Sharpening revisited

- What does blurring take away?

\[ \text{original} \quad - \quad \text{smoothed (5x5)} \quad = \quad \text{detail} \]

Let's add it back:

\[ \text{original} \quad + \quad \alpha \quad \text{detail} \quad = \quad \text{sharpened} \]
More on Linear Operations: Sharpening Filters

• Sharpening filters use masks that typically have + and – numbers in them.

• They are useful for highlighting or enhancing details and high-frequency information (e.g. edges)

• They can (and often are) based on derivative-type operations in the image (whereas smoothing operations were based on “integral” type operations)

Differentiation and convolution

• Recall, for 2D function, \( f(x,y) \):

\[
\frac{\partial f}{\partial x} = \lim_{\varepsilon \to 0} \left( \frac{f(x + \varepsilon, y) - f(x, y)}{\varepsilon} \right)
\]

• This is linear and shift invariant, so must be the result of a convolution.

• We could approximate this as

\[
\frac{\partial f}{\partial x} \approx \frac{f(x_{a+1}, y) - f(x_a, y)}{\Delta x}
\]

• which is obviously a convolution with kernel

\[
\begin{bmatrix}
-1 \\
1
\end{bmatrix}
\]

Source: D. Forsyth, D. Lowe
Derivative-type Filters

\[
\begin{bmatrix}
-1 & 1 \\
-1 & 1 \\
1 & -2 & 1 \\
1 & -2 & 1
\end{bmatrix}
\rightarrow 
\begin{align*}
\frac{\partial f}{\partial x} & \approx f(x+1, y) - f(x, y) \\
\frac{\partial f}{\partial y} & \approx f(x, y+1) - f(x, y) \\
\frac{\partial^2 f}{\partial x^2} & \approx f(x+1, y) - 2f(x, y) + f(x-1, y) \\
\frac{\partial^2 f}{\partial y^2} & \approx f(x, y+1) - 2f(x, y) + f(x, y-1)
\end{align*}
\]

Laplacian: \( \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \Rightarrow [1 \ -2 \ 1] + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \)

Variations of the Laplacian Filter

Laplacian: \( \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \Rightarrow [1 \ -2 \ 1] + \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \)

Same response in row/column directions

Consider: \( \begin{bmatrix} 1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 1 \\ 0 & -2 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix} \)

Same response in diagonal directions

Together: \( \begin{bmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \)

“Isotropic” filter
Sharpening Using the Laplacian Filter

\[ g(x, y) = A \cdot f(x, y) - \nabla^2 f(x, y) \quad \rightarrow \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & A+8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \]

Boosting High Frequencies
Gaussian Unsharp Mask Filter

\[ f + \alpha (f - f * g) = (1 + \alpha) f - \alpha f * g = f * ((1 + \alpha) e - g) \]

- **Goal:** Identify sudden changes (discontinuities) in an image
  - Intuitively, most semantic and shape information from the image can be encoded in the edges
  - More compact than pixels

- **Ideal:** artist's line drawing (but artist is also using object-level knowledge)

Source: D. Lowe
Characterizing edges

- An edge is a place of rapid change in the image intensity function along horizontal scanline.

The gradient of an image:

\[ \nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \end{bmatrix} \]

- The edge strength is given by the gradient magnitude.

The gradient points in the direction of most rapid increase in intensity.

- How does this direction relate to the direction of the edge?

The gradient direction is

\[ \theta = \tan^{-1} \left( \frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right) \]

The edge strength is given by the gradient magnitude

\[ \| \nabla f \| = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2} \]

Source: Steve Seitz
Using the first derivative for enhancement:

\[ \nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \Rightarrow \|\nabla f\| = \left(\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right)^{1/2} \approx \left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial f}{\partial y}\right| \]

- Sobel Edge Detector:
  - Given image \( f \) apply:
    
    \[
    \begin{bmatrix}
    -1 & 0 & 1 \\
    -2 & 0 & 2 \\
    -1 & 0 & 1
    \end{bmatrix} \approx \frac{\partial f}{\partial x}
    \begin{bmatrix}
    -1 & -2 & -1 \\
    0 & 0 & 0 \\
    1 & 2 & 1
    \end{bmatrix} \approx \frac{\partial f}{\partial y}
    \]

Application of Sobel gradient:

Many other edge detecting filters exist. Which is best?
Order-statistics Filters

- Linear filters of the type we have seen (with all positive coefficients) will blur the image and reduce certain kinds of noise.

- Nonlinear smoothing filters can also be considered
  - Instead of computing a weighted average over the masked area, perform an operation on the sorted list of pixels in the area.

- Order statistic filters are useful for removing certain “impulsive” types of noise.
Examples of Order-statistics Filters

Nonlinear (OS) Filter

\[
\begin{bmatrix}
  f_1 & f_2 & \cdots & f_8 & f_9 \\
\end{bmatrix}
\]

sort

\[
\begin{bmatrix}
  z_1 & z_2 & \cdots & z_8 & z_9 \\
\end{bmatrix}
\]

Inner product

\[
g = \sum_i w_i z_i = w^T z
\]

Median Filter

\[
\begin{bmatrix}
  0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

Min Filter

\[
\begin{bmatrix}
  1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

Max Filter

\[
\begin{bmatrix}
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

Trimmed Mean Filter

\[
\begin{bmatrix}
  0 & 0 & 0 & \frac{1}{4} & 1 & \frac{1}{4} & 0 & 0 & 0 \\
\end{bmatrix}
\]

• For Gaussian noise removal: Use linear smoothing filter

• For impulsive (Salt+Pepper, heavytailed,..): Use order statistic filter

**FIGURE 3.37** (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. Pascale, Lixi, Inc.)