Computer Vision
CSC5930-101
CSC9010-006

Instructor: Dr. Edward Kim
Histogram

Distribution of gray levels, how frequently each level occurs in the image
Histogram pseudo code
Image filtering

• Image filtering: compute function of local neighborhood at each position

• Really important!
  – Enhance images
    • Denoise, resize, increase contrast, etc.
  – Extract information from images
    • Texture, edges, distinctive points, etc.
  – Detect patterns
    • Template matching
Instagram Filters

No filter  Vignette  Black & white
Warm  Cool  Vintage
Example: box filter

\[ g[\cdot, \cdot] \]

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

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

Slide credit: David Lowe (UBC)
Image filtering

\[ f[\ldots] \]

\[ h[\ldots] \]

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]
Image filtering

\[ f[\ldots] \]

\[ h[\ldots] \]

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]

Credit: S. Seitz
Image filtering

\[ f[\cdot,\cdot] \]

\[ h[\cdot,\cdot] \]

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]

Credit: S. Seitz
Image filtering

\[ f\left[ \cdots \right] \]

\[ h\left[ \cdots \right] \]

\[
\begin{align*}
\frac{1}{9} & \begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1
\end{bmatrix}
\end{align*}
\]

\[
h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l]
\]

Credit: S. Seitz
Image filtering

\[ f[\ldots] \quad h[\ldots] \]

\[
h[m,n] = \sum_{k,l} g[k,l] \, f[m+k,n+l]
\]

Credit: S. Seitz
Image filtering

\[
f[\ldots] \quad h[\ldots]
\]

\[
h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l]
\]

Credit: S. Seitz
Image filtering

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l] \]

Credit: S. Seitz
Image filtering

\[ f[\ldots] \]

\[ h[\ldots] \]

\[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]

Credit: S. Seitz
Box Filter

What does it do?

• Replaces each pixel with an average of its neighborhood

• Achieve smoothing effect (remove sharp features)

Slide credit: David Lowe (UBC)
Smoothing with box filter
Practice with linear filters

Original

Source: D. Lowe
Practice with linear filters

Original

Filtered
(no change)

Source: D. Lowe
Practice with linear filters

Original

Source: D. Lowe
Practice with linear filters

Original

Shifted left
By 1 pixel

Source: D. Lowe
Other filters

• Brightness filter?
• Blurring filter?
• Sharpening Filter?
• Gradient Filter??
Gradient

• Derivative of an image
  – Rate of change
  – Speed is the rate of change of distance
  – Acceleration is the rate of change of speed
\[
\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x) = f_x
\]
\[ y = x^2 + x^4 \]
\[ y = x^2 + x^4 \]

\[ \frac{dy}{dx} = 2x + 4x^3 \]
Discrete Derivatives

\[
\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x)
\]
Discrete Derivatives

\[
\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x)
\]

\[
\frac{df}{dx} = \frac{f(x) - f(x - 1)}{1} = f'(x)
\]
\[
\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x)
\]

\[
\frac{df}{dx} = \frac{f(x) - f(x - 1)}{1} = f''(x)
\]

\[
\frac{df}{dx} = f(x) - f(x - 1) = f'(x)
\]


\[
\frac{df}{dx} = f(x) - f(x-1) = f'(x)
\]

Backward difference
Finite Differences

\[
\frac{df}{dx} = f(x) - f(x - 1) = f'(x)
\]
Backward difference

\[
\frac{df}{dx} = f(x) - f(x + 1) = f'(x)
\]
Forward difference
Finite Differences

\[ \frac{df}{dx} = f(x) - f(x - 1) = f'(x) \quad \text{Backward difference} \]

\[ \frac{df}{dx} = f(x) - f(x + 1) = f'(x) \quad \text{Forward difference} \]

\[ \frac{df}{dx} = f(x + 1) - f(x - 1) = f'(x) \quad \text{Central difference} \]
Example

\[ f(x) = 10 \quad 15 \quad 10 \quad 10 \quad 25 \quad 20 \quad 20 \quad 20 \]
Example

\[ f(x) = 10 \ 15 \ 10 \ 10 \ 25 \ 20 \ 20 \ 20 \ 20 \]
\[ f'(x) = 0 \ 5 \ -5 \ 0 \ 15 \ -5 \ 0 \ 0 \ 0 \]
Example

\[ f(x) = 10 \quad 15 \quad 10 \quad 10 \quad 25 \quad 20 \quad 20 \quad 20 \quad 20 \]

\[ f'(x) = 0 \quad 5 \quad -5 \quad 0 \quad 15 \quad -5 \quad 0 \quad 0 \quad 0 \]

\[ f''(x) = 0 \quad 5 \quad -10 \quad 5 \quad 15 \quad -20 \quad 5 \quad 0 \quad 0 \]
Derivatives in 2D

Given function

\[ f(x, y) \]

Gradient vector

\[ \nabla f(x, y) = \begin{bmatrix} \frac{\partial f(x, y)}{\partial x} \\ \frac{\partial f(x, y)}{\partial y} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix} \]

Gradient magnitude

\[ |\nabla f(x, y)| = \sqrt{f_x^2 + f_y^2} \]
Derivatives of Images

Derivative masks

\[ f_x \Rightarrow \frac{1}{3} \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad f_y \Rightarrow \frac{1}{3} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \]
Derivatives of Images

Derivative masks

\[ f_x \Rightarrow \frac{1}{3} \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad f_y \Rightarrow \frac{1}{3} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \]

\[
I = \begin{bmatrix}
10 & 10 & 20 & 20 & 20 \\
10 & 10 & 20 & 20 & 20 \\
10 & 10 & 20 & 20 & 20 \\
10 & 10 & 20 & 20 & 20 \\
10 & 10 & 20 & 20 & 20
\end{bmatrix}
\]
Derivatives of Images

Derivative masks

\[ f_x \Rightarrow \frac{1}{3} \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad f_y \Rightarrow \frac{1}{3} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \]

\[ I = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix} \]
Derivatives of Images

Derivative masks

\[ f_x \Rightarrow \frac{1}{3} \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad f_y \Rightarrow \frac{1}{3} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \]

\[
I = \begin{bmatrix}
10 & 10 & 20 & 20 & 20 \\
10 & 10 & 20 & 20 & 20 \\
10 & 10 & 20 & 20 & 20 \\
10 & 10 & 20 & 20 & 20 \\
10 & 10 & 20 & 20 & 20 \\
\end{bmatrix}
\]

\[
I_x = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 10 & 10 & 0 & 0 \\
0 & 10 & 10 & 0 & 0 \\
0 & 10 & 10 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\]
Derivatives of Images

Derivative masks

\[ f_x \Rightarrow \frac{1}{3} \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \quad f_y \Rightarrow \frac{1}{3} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \]

\[ I = \begin{bmatrix} 10 & 10 & 20 & \text{red} & \text{red} & \\ 10 & 10 & 20 & \text{red} & \text{red} & \\ 10 & 10 & 20 & \text{red} & \text{red} & \\ 10 & 10 & 20 & \text{red} & \text{red} & \\ 10 & 10 & 20 & \end{bmatrix} \quad I_x = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 10 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]
Derivatives of Images

\[ I = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix} \quad \quad I_y = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]
Other filters

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>0</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>-2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-1</td>
<td></td>
</tr>
</tbody>
</table>

Sobel

Vertical Edge
(absolute value)
Other filters

Sobel

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-1</td>
<td>-2</td>
<td>-1</td>
</tr>
</tbody>
</table>

Horizontal Edge (absolute value)
Let $F(x,y)$ be the 7x7 neighborhood, $H(u,v)$ be the convolution kernel, and $G(x,y)$ be the result of the convolution.

$$F(x,y) \times H(u,v) = G(x,y)$$
\[ F(x,y) \times H(u,v) = G(x,y) \]
$F(x,y)$ \times \begin{array}{ccc}
    a & b & c \\
    d & e & f \\
    g & h & i \\
\end{array} = \begin{array}{c}
    0 \\
    f \\
    e \\
    d \\
\end{array}$ 

$G(x,y)$
Filtering vs. Convolution

• 2d filtering
  \[ h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l] \]
  - \( h = \text{filter2}(g,f); \) or
  \( h = \text{imfilter}(f,g); \)

• 2d convolution
  \[ h[m,n] = \sum_{k,l} g[k,l] f[m-k,n-l] \]
  - \( h = \text{conv2}(g,f); \)
properties

- **Commutative:** \( a * b = b * a \)
  - Conceptually no difference between filter and signal
  - But particular filtering implementations might break this equality

- **Associative:** \( a * (b * c) = (a * b) * c \)
  - Often apply several filters one after another: \(((a * b_1) * b_2) * b_3)\)
  - This is equivalent to applying one filter: \(a * (b_1 * b_2 * b_3)\)

- **Distributes over addition:** \( a * (b + c) = (a * b) + (a * c) \)

- **Scalars factor out:** \( ka * b = a * kb = k (a * b) \)

- **Identity:** unit impulse \( e = [0, 0, 1, 0, 0] \), \( a * e = a \)

Source: S. Lazebnik
Gaussian

\[ g(x) = e^{\frac{-x^2}{2\sigma^2}} \]
Important filter: Gaussian

- Weight contributions of neighboring pixels by nearness

$$G_\sigma = \frac{1}{2\pi \sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.003</td>
<td>0.013</td>
<td>0.022</td>
<td>0.013</td>
<td>0.003</td>
</tr>
<tr>
<td>0.013</td>
<td>0.059</td>
<td>0.097</td>
<td>0.059</td>
<td>0.013</td>
</tr>
<tr>
<td>0.022</td>
<td>0.097</td>
<td>0.159</td>
<td>0.097</td>
<td>0.022</td>
</tr>
<tr>
<td>0.013</td>
<td>0.059</td>
<td>0.097</td>
<td>0.059</td>
<td>0.013</td>
</tr>
<tr>
<td>0.003</td>
<td>0.013</td>
<td>0.022</td>
<td>0.013</td>
<td>0.003</td>
</tr>
</tbody>
</table>

$5 \times 5, \sigma = 1$

Slide credit: Christopher Rasmussen
Smoothing with Gaussian filter
Smoothing with box filter
Box vs Gaussian

box filter

 gaussian
Gaussian filters

- Remove “high-frequency” components from the image (low-pass filter)
  - Images become more smooth
- 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

Source: K. Grauman
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.

Source: D. Lowe
Separability example

2D convolution (center location only)

\[
\begin{array}{ccc}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1 \\
\end{array}
\quad \ast 
\begin{array}{ccc}
2 & 3 & 3 \\
3 & 5 & 5 \\
4 & 4 & 6 \\
\end{array}
\]

The filter factors into a product of 1D filters:

\[
\begin{array}{ccc}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1 \\
\end{array}
\quad \ast 
\begin{array}{ccc}
1 \\
2 \\
1 \\
\end{array}
\quad \times 
\begin{array}{ccc}
1 & 2 & 1 \\
\end{array}
\]

Perform convolution along rows:

\[
\begin{array}{ccc}
1 & 2 & 1 \\
\end{array}
\quad \ast 
\begin{array}{ccc}
2 & 3 & 3 \\
3 & 5 & 5 \\
4 & 4 & 6 \\
\end{array}
\]

Followed by convolution along the remaining column:

\[
\begin{array}{ccc}
\end{array}
\quad \ast 
\begin{array}{ccc}
\end{array}
\]

Source: K. Grauman
Separability

• Why is separability useful in practice?
Practical matters

How big should the filter be?

• Values at edges should be near zero
• Rule of thumb for Gaussian: set filter half-width to about $3 \sigma$
Practical matters

• 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
Median filters

• A **Median Filter** operates over a window by selecting the median intensity in the window.

• What advantage does a median filter have over a mean filter?
Comparison: salt and pepper noise

<table>
<thead>
<tr>
<th>3x3</th>
<th>5x5</th>
<th>7x7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>Gaussian</td>
<td>Median</td>
</tr>
</tbody>
</table>

Slide by Steve Seitz
Morphology

Figure 3.21  Binary image morphology: (a) original image; (b) dilation; (c) erosion; (d) majority; (e) opening; (f) closing. The structuring element for all examples is a $5 \times 5$ square. The effects of majority are a subtle rounding of sharp corners. Opening fails to eliminate the dot, since it is not wide enough.
\[ \theta(f, t) = \begin{cases} 
1 & \text{if } f \geq t, \\
0 & \text{else}, 
\end{cases} \]

\[ c = f \otimes s \]

\(c\) is the count of the number of 1’s as the result of \(f\) convolved with \(s\).

- **dilation**: \( \text{dilate}(f, s) = \theta(c, 1) \);
- **erosion**: \( \text{erode}(f, s) = \theta(c, S) \);
- **majority**: \( \text{maj}(f, s) = \theta(c, S/2) \);
- **opening**: \( \text{open}(f, s) = \text{dilate}(\text{erode}(f, s), s) \);
- **closing**: \( \text{close}(f, s) = \text{erode}(\text{dilate}(f, s), s) \).
Practical matters

– methods (MATLAB):
  • clip filter (black): \texttt{imfilter(f, g, 0)}
  • wrap around: \texttt{imfilter(f, g, ‘circular’ )}
  • copy edge: \texttt{imfilter(f, g, ‘replicate’ )}
  • reflect across edge: \texttt{imfilter(f, g, ‘symmetric’ )}

Source: S. Marschner