Basic Filters (7)

- Convolution/correlation/Linear filtering
- Gaussian filters
- Smoothing and noise reduction
- First derivatives of Gaussian
- Second derivative of Gaussian: Laplacian
- Oriented Gaussian filters
- Steerability
Convolution

• 1D Formula:
  \[(h * f)(x) = \int h(x - u) f(u) du\]

  \[(h * f)[x] = \sum_i h[x - i] f[i]\]

• 2D Formula:
  \[(h * f)(x, y) = \int h(x - u, y - v) f(u, v) du\]
  \[\text{Should integrate over } v, \text{ too!}\]

  \[(h * f)[x, y] = \sum_i h[x - i, y - j] f[i, j]\]
  \[\text{Should sum over } j, \text{ too!}\]

• Example on the web:
Slight Blurring

Kernel:

<table>
<thead>
<tr>
<th>1/9</th>
<th>1/9</th>
<th>1/9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/9</td>
<td>1/9</td>
<td>1/9</td>
</tr>
<tr>
<td>1/9</td>
<td>1/9</td>
<td>1/9</td>
</tr>
</tbody>
</table>
Kernel:
15 x 15 matrix of value 1/225
Basic Properties

- Commutes: \( f \ast g = g \ast f \)
- Associative: \( (f \ast g) \ast h = f \ast (g \ast h) \)
- Linear: \( (af + bg) \ast h = a \ast f \ast h + b \ast g \ast h \)
- Shift invariant: \( f_t \ast h = (f \ast h)_t \)
- Only operator both linear and shift invariant
- Differentiation: \[ \frac{\partial}{\partial x} \left( f \ast g \right) = \frac{\partial f}{\partial x} \ast g \]
Practicalities (discrete convolution/correlation)

- **MATLAB**: `conv` (1D) or `conv2` (2D), `corr`
- **Border issues**:  
  - When applying convolution with a $K \times K$ kernel, the result is undefined for pixels closer than $K$ pixels from the border of the image
- **Options**:  
  - Warp around
  - Expand/Pad
  - Crop
1-D:
\[ g(x) = e^{-\frac{x^2}{2\sigma^2}} \]

2-D:
\[ G(x, y) = e^{-\frac{x^2 + y^2}{2\sigma^2}} \]

Slight abuse of notations:
We ignore the normalization constant such that
\[ \int g(x) \, dx = 1 \]
Simple Averaging

Gaussian Smoothing
Image Noise

\[ f(x, y) = \hat{f}(x, y) + \eta(x, y) \]

Gaussian i.i.d. ("white") noise:
\[ \eta(x, y) \sim \mathcal{N}(\mu, \sigma) \]
Gaussian Smoothing to Remove Noise

No smoothing

\[ \sigma = 2 \]

\[ \sigma = 4 \]
Shape of Gaussian filter as function of $\sigma$
Note about Finite Kernel Support

- Gaussian function has infinite support

- In discrete filtering, we have finite kernel

\[ \sigma = 5 \text{ with } 10\times10 \text{ kernel} \]

\[ \sigma = 5 \text{ with } 30\times30 \text{ kernel} \]
Basic Properties

- Gaussian removes “high-frequency” components from the image → “low pass” filter
- Larger \( \sigma \) remove more details
- Combination of 2 Gaussian filters is a Gaussian filter:
  \[
  G_{\sigma_1} \ast G_{\sigma_2} = G_{\sigma} \quad \sigma^2 = \sigma_1^2 + \sigma_2^2
  \]
- Separable filter:
  \[
  G_{\sigma} \ast f = g_{\sigma} \rightarrow \ast g_{\sigma} \uparrow \ast f
  \]
- Critical implication: Filtering with a \( N \times N \) Gaussian kernel can be implemented as two convolutions of size \( N \) → reduction quadratic to linear → \textit{must} be implemented that way
Image Derivatives

- Image Derivatives
- Derivatives increase noise
- Derivative of Gaussian
- Laplacian of Gaussian (LOG)
Image Derivatives

• We want to compute, at each pixel \((x,y)\) the derivatives:

• In the discrete case we could take the difference between the left and right pixels:

\[
\frac{\partial I}{\partial x} \approx I(i + 1, j) - I(i - 1, j)
\]

Note: might want to divide by 2

• Convolution of the image by

\[
\partial_x = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}
\]

• Problem: Increases noise

\[
I(i + 1, j) - I(i - 1, j) = \hat{I}(i + 1, j) - \hat{I}(i - 1, j) + n_+ + n_-
\]

Difference between Actual image values

True difference (derivative)

Twice the amount of noise as in the original image
Smooth Derivatives

• Solution: First smooth the image by a Gaussian $G_\sigma$ and then take derivatives:
  \[
  \frac{\partial f}{\partial x} \approx \frac{\partial (G_\sigma * f)}{\partial x}
  \]

• Applying the differentiation property of the convolution:
  \[
  \frac{\partial f}{\partial x} \approx \frac{\partial G_\sigma}{\partial x} * f
  \]

• Therefore, taking the derivative in $x$ of the image can be done by convolution with the derivative of a Gaussian:
  \[
  G_\sigma^x = \frac{\partial G_\sigma}{\partial x} = xe^{-\frac{x^2+y^2}{2\sigma^2}}
  \]

• Crucial property: The Gaussian derivative is also separable:
  \[
  G_\sigma^x * f = g_\sigma^x * g_\sigma^\uparrow * f
  \]
Derivative + Smoothing

Better but still blurs away edge information
Applying the first derivative of Gaussian

\[ |\nabla I| = \sqrt{\frac{\partial I^2}{\partial x} + \frac{\partial I^2}{\partial y}} \]
There is ALWAYS a tradeoff between smoothing and good edge localization!

- Image with Edge
- Edge Location

- Image + Noise
- Derivatives detect edge and noise
- Smoothed derivative removes noise, but blurs edge
Second derivatives:
Laplacian

\[ \nabla^2 G = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \]

Either left G should be I or right I’s should be G’s
DOG Approximation to LOG

$$\nabla^2 G_\sigma \approx G_{\sigma_1} - G_{\sigma_2}$$
Separable, low-pass filter

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

Gaussian

\[
\frac{\partial G_\sigma(x, y)}{\partial x} \propto xe^{-\frac{x^2+y^2}{2\sigma^2}} \quad \frac{\partial G_\sigma(x, y)}{\partial y} \propto ye^{-\frac{x^2+y^2}{2\sigma^2}}
\]

Separable, output of convolution is gradient at scale \( \sigma \):

\[ \nabla I = I \ast \nabla G_\sigma \]

Derivatives of Gaussian

\[ \nabla G_\sigma = \begin{bmatrix} \frac{\partial G_\sigma}{\partial x} & \frac{\partial G_\sigma}{\partial y} \end{bmatrix}^t \]

Laplacian

\[ \nabla^2 G_\sigma(x, y) = \frac{\partial^2 G_\sigma(x, y)}{\partial x^2} + \frac{\partial^2 G_\sigma(x, y)}{\partial y^2} \]

Not-separable, approximated by A difference of Gaussians. Output of convolution is Laplacian of image: Zero-crossings correspond to edges

Directional Derivatives

\[ \cos \theta \frac{\partial G_\sigma}{\partial x} + \sin \theta \frac{\partial G_\sigma}{\partial y} \]

Output of convolution is magnitude of derivative in direction \( \theta \). Filter is linear combination of derivatives in \( x \) and \( y \)

Oriented Gaussian

\[ e^{-\frac{(a_1x+b_1y)^2}{2\sigma_1^2}} \quad e^{-\frac{(a_2x+b_2y)^2}{2\sigma_2^2}} \]

Smooth with different scales in orthogonal directions