Region Processing III

Prof. Eric Miller
elmiller@ece.tufts.edu
Finding edges

• Another very common task in image processing is finding edges in images
• Ideal 1D edge

• Want a filter that outputs a large value only where the edge is. Something like

• What can we use?
2D edges

• In 2D we have lots more possibilities for edges

• What can we do??
• Here: stick with horizontal and vertical edges
1D Edges

Model of an edge

\[ e(t) = \tan^{-1}(10t) \]

Derivative of the edge

\[ \frac{de}{dt} = \frac{1}{1 + t^2} \]

Edge = peak in first derivative

Second derivative of the edge

\[ \frac{d^2e}{dt^2} = -2 \frac{t}{(1 + t^2)^2} \]

Edge = zero crossing in second derivative
Edge Filtering

- Most edge filters in 2D operate on the same ideas
- 2D first derivative = gradient
  \[ \nabla f(x, y) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \]
  - Vertical edges
  - Horizontal edges
- 2D second derivative = Laplacian
  \[ \nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \]
Discrete Derivatives = Filters

\[ \frac{\partial f}{\partial x} \approx \frac{f(x+\delta, y) - f(x, y)}{\delta} \]

- With \( y \) fixed, the partial derivative with respect to \( x \) is just the difference between \( f \) at \( x \) and \( f \) at \( x \) plus a little.
- In a digital image, we can only move over by discrete pixels so the digital version of the above is:

\[
df_v[m, n] = f[m + 1, n] - f[m, n] = f[m, n] * h_v[m, n]
\]

- Similarly, for horizontal edges we would have:

\[
df_h[m, n] = f[m, n + 1] - f[m, n] = f[m, n] * h_h[m, n]
\]

\[ h_v[m, n] \]

\[ h_h[m, n] \]
Less Simple Filters

- In practice, these simple filters are not used.
- Slightly more sophisticated ones have been developed.

- Filters provide some smoothing in direction perpendicular to the edge.
- Why? See Matlab.
Laplacian filters

Second derivative is first derivative done twice
Laplacian again

- Same problems as before: smoothing tends to help a bit. So, other 3x3 filters also used to approximate the Laplacian.
Final Laplacian

- One last common idea is to first smooth the image with Gaussian and then find Laplacian

\[
\begin{align*}
  f & \rightarrow \ast g(x, y) \rightarrow \nabla^2 \\
  f & \rightarrow \ast \left[ \nabla^2 g(x, y) \right]
\end{align*}
\]

- Smoothing and then finding the Laplacian is the same as smoothing with the Laplacian of a Gaussian (LoG)

- This type of mixing of operations cannot be done in general. Systems have to be linear and shift invariant (more on this junior year)
LoG Filter

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

\[\left( \frac{2}{\sigma^2} - \frac{x^2 + y^2}{\sigma^4} \right) \exp \left( -\frac{1}{2\sigma^2} (x^2 + y^2) \right)\]

Note: 4 to 1 peak contrast like discrete case
Matlab Topics

• Use of `meshgrid` for generating sets of 2D points
• Use of `fspecial` for building filters
• Use of `mesh` for displaying surfaces and `imagesc` as an alternate for `imshow` for displaying pictures