Colorado School of Mines

Computer Vision

Professor William Hoff
Dept of Electrical Engineering & Computer Science
http://inside.mines.edu/~whoff/
Image Filtering
Image Filtering

• We will briefly look at methods to
  – Reduce noise and enhance images
  – Detect features
  – (These topics are covered in more detail in “image processing”)

• Topics
  – Gray level (point transforms)
  – Spatial (neighborhood) transforms
  – Binary image processing
Gray Level Transformations

• Point operations
  – \( s = f(r) \)
  – Map input pixel value \( r \) to output value \( s \)

• Examples

**Linear scaling**

\[ s = cr \]

**Square** (\( s = cr^2 \))

Enhances contrast of high intensities

**Log** (\( s = c \text{ log}(1+r) \))

Enhances contrast of low intensities
Gamma Correction

\[ s = cr^\gamma \]

FIGURE 3.6 Plots of the equation \( s = cr^\gamma \) for various values of \( \gamma \) (\( c = 1 \) in all cases).
FIGURE 3.9
(a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 3.0, 4.0, \text{ and } 5.0$, respectively. (Original image for this example courtesy of NASA.)
Demos

• Matlab
  – Load image pout.tif
    • Sample images are in the Matlab folder under “toolbox\image\imdemos”
  – See histogram
    • imhist
  – Adjust limits (linear scaling)
    • imtool

• Photoshop
  – Can draw arbitrary transformation curve
  – Image->adjustments->curves ...
Histogram Equalization

- Think of the histogram $H(r)$ as the (scaled) probability distribution of the input image values.
- Want histogram of the output image to be flat: $p(s) = \text{const}$

\[ H(r) \]
\[ \text{or} \]
\[ p(r) \]

\[ H(s) \]
\[ \text{or} \]
\[ p(s) \]

- This stretches contrast where the original image had many pixels with a certain range of gray levels and compresses contrast elsewhere.
The desired mapping function is just the cumulative probability distribution function of the input image

\[ P(r) = \int_0^r p_r(w) \, dw \]
Examples

• Matlab histogram equalization
  – histeq(I);
  – To see transform function
    • [I2,T]=histeq(I);
    • plot(T)
• Adaptive histogram equalization
  – Apply to local neighborhoods
  – adapthisteq(I);

Try image “liftingbody.png”

Input image “liftingbody.png”  Histogram equalization  Adaptive histogram equalization
Spatial Filtering

- Filter or mask \( w \), size \( m \times n \)
- Apply to image \( f \), size \( M \times N \)
- Sum of products of mask coeffs with corresponding pixels under mask
- Slide mask over image, apply at each point
- Also called “cross-correlation”

\[
g(x, y) = \sum_{s=-m/2}^{m/2} \sum_{t=-n/2}^{n/2} w(s, t) \ f(x + s, y + t) \\
= w(x, y) \otimes f(x, y)
\]
Example

- Box or averaging filter
- Can use to smooth image (blur, remove noise)
- Manual calculation on corner image?

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

\[
\frac{1}{9} \times \begin{array}{ccc}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{array}
\]

\[
\frac{1}{16} \times \begin{array}{ccc}
1 & 2 & 1 \\
2 & 4 & 2 \\
1 & 2 & 1 \\
\end{array}
\]

Remember to divide by 9 (or 16)
Example

- Box or averaging filter
- Can use to smooth image (blur, remove noise)
- Manual calculation on corner image?

\[
\begin{array}{|c|c|c|}
\hline
1 & 1 & 1 \\
\hline
\end{array}
\quad \frac{1}{9} \times \quad \begin{array}{|c|c|c|}
\hline
1 & 1 & 1 \\
\hline
\end{array}
\quad \frac{1}{16} \times \quad \begin{array}{|c|c|c|}
\hline
1 & 2 & 1 \\
\hline
\end{array}
\quad \begin{array}{|c|c|c|}
\hline
1 & 1 & 1 \\
\hline
\end{array}
\quad 2 & 4 & 2 \\
\hline
1 & 2 & 1 \\
\hline
\end{array}
\]

*Remember to divide by 9 (or 16)*
FIGURE 3.35 (a) Original image, of size $500 \times 500$ pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $n = 3, 5, 9, 15, 25, 35$, respectively. The black squares at the top are of sizes $3, 5, 9, 15, 25, 35, 45, 55$ pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their gray levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size $50 \times 120$ pixels.
Matlab Examples

- **fspecial** - useful to create filters
- **imfilter** - to apply to an image

```matlab
>> clear all
>> close all
>> I = zeros(200,200);
>> I(50:150, 50:150) = 1;
>> imshow(I,[],
>> w = [ 1 1 1; 1 1 1; 1 1 1]/9

w =

0.1111 0.1111 0.1111
0.1111 0.1111 0.1111
0.1111 0.1111 0.1111

>> I2 = imfilter(I, w);
>> imshow(I2,[],
>> w = fspecial('average', [5 5])
>> I3 = imfilter(I, w);
>> imshow(I3,[],
>> w = fspecial('average', [15 15])
```
Gaussian Smoothing Filter

• Gaussian filter usually preferable to box filter
• Attenuates high frequencies better

$$h(x, y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/(2\sigma^2)}$$
Convolutions vs Correlations

- Cross-correlation of mask \( h(x, y) \) with image \( f(x, y) \)

\[
g(x, y) = \sum_{s=-m/2}^{m/2} \sum_{t=-n/2}^{n/2} h(s, t) f(x + s, y + t) = h \otimes f
\]

- Convolution of mask \( h(x, y) \) with image \( f(x, y) \)

\[
g(x, y) = \sum_{s=-m/2}^{m/2} \sum_{t=-n/2}^{n/2} h(s, t) f(x - s, y - t) = h \ast f
\]

- or

\[
g(x, y) = \sum_{s=-m/2}^{m/2} \sum_{t=-n/2}^{n/2} f(x, y) h(x - s, y - t) = f \ast h
\]

- Convolution same as correlation except that we first flip one function about the origin
Sharpening Spatial Filters

- First derivative (can also do central difference)
  \[ \frac{\partial f}{\partial x} \approx f(x+1) - f(x) \]

- Second derivative
  \[ \frac{\partial^2 f}{\partial x^2} \approx f(x+1) - 2f(x) + f(x-1) \]
Edge Detection

Smoothed step edge

First derivative \( \frac{\partial f}{\partial x} \)

Second derivative \( \frac{\partial^2 f}{\partial x^2} \)

Peak magnitude at location of edge

Zero crossing at location of edge
**Edge Operators for 2D Images**

<table>
<thead>
<tr>
<th>$\frac{\partial}{\partial x}$</th>
<th>$\frac{\partial}{\partial y}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1 0 +1</td>
<td>-1 2 -1</td>
</tr>
<tr>
<td>-2 0 +2</td>
<td>0 0 0</td>
</tr>
<tr>
<td>-1 0 +1</td>
<td>+1 +2 +1</td>
</tr>
</tbody>
</table>

*Sobel operators*

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

| 0 1 0 |
| 1 -4 1 |
| 0 1 0 |

*Laplacian operator*

- Example:
  - Manual calculation of Sobel on corner
  - Compare correlation vs convolution
Matlab Examples

• Try image “moon.tif”
• Create Sobel masks
  – \(hx = \begin{bmatrix} -1 & 0 & 1; -2 & 0 & 2; -1 & 0 & 1 \end{bmatrix}; hy = hx';\)
• imfilter to do correlation
  – See difference if convert image to double first
  – \(I=\text{double}(I); \) % just changes type
  – \(I=\text{im2double}(I); \) % change type and scale to 0..1
• Notes
  – \texttt{filter2} – same as \texttt{imfilter} but always converts to double
  – \texttt{conv2} – does convolution
Gradient

• Compute gradient components using first derivative operators

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

• Gradient magnitude shows location of edges in the image

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

• Gradient angle shows direction of edge

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

• Compute gradient components using first derivative operators
  – \( \text{Dx} = \text{imfilter}(I, \text{hx}) \)

• Gradient magnitude peaks at locations of edges in the image
  – \( (\text{Dx} \cdot \text{Dx} + \text{Dy} \cdot \text{Dy}) \cdot 0.5 \)

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

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

Note – the period in Matlab indicates a point-by-point operation instead of a matrix operation

• Gradient angle shows direction of edge
  – \( \text{atan2} (\text{Dy}, \text{Dx}) \)
  – colormap jet
  – colorbar

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