Motivation

- Noise reduction/image restoration

- Structure extraction

Common Types of Noise

- Salt & pepper noise
  - Random occurrences of black and white pixels

- Impulse noise
  - Random occurrences of white pixels

- Gaussian noise
  - Variations in intensity drawn from a Gaussian ("Normal") distribution.

- Basic Assumption
  - Noise is i.i.d. (independent & identically distributed)

Gaussian Noise

- Linear Filters
  - What are they? How are they applied?
  - Application: smoothing
  - Gaussian filter
  - What does it mean to filter an image?

- Nonlinear Filters
  - Median filter

- Multi-Scale representations
  - How to properly rescale an image?

- Image derivatives
  - How to compute gradients robustly?
First Attempt at a Solution

- Assumptions:
  - Expect pixels to be like their neighbors
  - Expect noise processes to be independent from pixel to pixel ("i.i.d. = independent, identically distributed")
- Let’s try to replace each pixel with an average of all the values in its neighborhood…

Moving Average in 2D

\[ F[x, y] \quad G[x, y] \]
Correlation Filtering

\[ G[i, j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u, v] F[i + u, j + v] \]

- This is called cross-correlation, denoted \( G = H \otimes F \).
- Filtering an image
  - Replace each pixel by a weighted combination of its neighbors.
  - The filter “kernel” or “mask” is the prescription for the weights in the linear combination.

Convolution vs. Correlation

- Correlation
  \[ G[i, j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u, v] F[i + u, j + v] \]
  \[ G = H \otimes F \]
- Convolution
  \[ G[i, j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u, v] F[i - u, j - v] \]
  \[ G = H \ast F \]
- Note
  \[ H H[-u,-v] = H[u,v], \text{ then correlation = convolution.} \]

Shift Invariant Linear System

- Shift invariant:
  - Operator behaves the same everywhere, i.e. the value of the output depends on the pattern in the image neighborhood, not the position of the neighborhood.
- Linear:
  - Superposition: \( h \ast (f_1 + f_2) = (h \ast f_1) + (h \ast f_2) \)
  - Scaling: \( h \ast (k f) = k (h \ast f) \)
Properties of Convolution

- Linear & shift invariant
- Commutative: \( f \ast g = g \ast f \)
- Associative: \( (f \ast g) \ast h = f \ast (g \ast h) \)
  - Often apply several filters in sequence: \( ((a \ast b_1) \ast b_2) \ast b_3 \)
  - This is equivalent to applying one filter: \( a \ast (b_1 \ast (b_2 \ast b_3)) \)
- Identity: \( f \ast e = f \)
  - for unit impulse \( e = [\ldots, 0, 0, 1, 0, 0, \ldots] \).
- Differentiation: \( \frac{\partial}{\partial x} (f \ast g) = \frac{\partial f}{\partial x} \ast g \)

Averaging Filter

- What values belong in the kernel \( H[u,v] \) for the moving average example?

\[
F[x, y] \ast H[u, v] = G[x, y]
\]

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

"box filter"

Smoothing by Averaging

Averaging Filter

Smoothing with a Gaussian

Gaussian Smoothing

- Gaussian kernel
  \[
  G_\sigma = \frac{1}{2\pi\sigma^2} \exp\left(\frac{-x^2}{2\sigma^2}\right)
  \]
- Rotationally symmetric
- Weights nearby pixels more than distant ones
  - This makes sense as ‘probabilistic’ inference about the signal
- A Gaussian gives a good model of a fuzzy blob
Gaussian Smoothing

- What parameters matter here?
- Variance $\sigma$ of Gaussian
  - Determines extent of smoothing

\[
\sigma = 2 \text{ with } 30 \times 30 \text{ kernel}
\]
\[
\sigma = 5 \text{ with } 30 \times 30 \text{ kernel}
\]

Slide credit: Kristen Grauman

Gaussian Smoothing

- What parameters matter here?
- Size of kernel or mask
  - Gaussian function has infinite support, but discrete filters use finite kernels

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

\[
\sigma = 5 \text{ with } 10 \times 10 \text{ kernel}
\]
\[
\sigma = 5 \text{ with } 30 \times 30 \text{ kernel}
\]

Slide credit: Kristen Grauman

Gaussian Smoothing in Matlab

```matlab
>> hsize = 10;
>> sigma = 5;
>> h = fspecial('gaussian', hsize, sigma);
>> mesh(h);
>> imagesc(h);
>> outim = imfilter(im, h);
>> imshow(outim);
```

Slide credit: Kristen Grauman

Effect of Smoothing

- More noise

Effect of Smoothing

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

Efficient Implementation

- Both, the BOX filter and the Gaussian filter are separable:
  - First convolve each row with a 1D filter
    \[
    g(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-x^2/(2\sigma^2))
    \]
  - Then convolve each column with a 1D filter
    \[
    g(y) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-y^2/(2\sigma^2))
    \]
- Remember:
  - Convolution is linear - associative and commutative
    \[
    g_x * g_y * f = g_y * (g_x * f) = (g_x * g_y) * f
    \]

Slide credit: Bernt Schiele

Filtering: Boundary Issues

- What is the size of the output?
- MATLAB: `imfilter2(g, f, shape)`
  - shape = ‘full’: output size is sum of sizes of f and g
  - shape = ‘same’: output size is same as f
  - shape = ‘valid’: output size is difference of sizes of f and g
Filtering: Boundary Issues

• How should the filter behave near the image boundary?
  ▶ The filter window falls off the edge of the image
  ▶ Need to extrapolate
  ▶ Methods:
    - Clip filter (black)
    - Wrap around
    - Copy edge
    - Reflect across 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

Topics of This Lecture

• Linear filters
  ▶ What are they? How are they applied?
  ▶ Application: smoothing
  ▶ Gaussian filter
  ▶ What does it mean to filter an image?

• Nonlinear Filters
  ▶ Median filter

• Multi-Scale representations
  ▶ How to properly rescale an image?

• Image derivatives
  ▶ How to compute gradients robustly?

Why Does This Work?

A small excursion into the Fourier transform to talk about spatial frequencies...

\[ 3 \cos(x) + 1 \cos(3x) + 0.8 \cos(5x) + 0.4 \cos(7x) + \ldots \]

Source: Michal Irani

The Fourier Transform in Pictures

A small excursion into the Fourier transform to talk about spatial frequencies...

\[ \begin{align*}
  A & \quad \Rightarrow \quad 3 \cos(x) \\
  B & \quad \Rightarrow \quad 1 \cos(3x) \\
  C & \quad \Rightarrow \quad 0.8 \cos(5x) \\
  D & \quad \Rightarrow \quad 0.4 \cos(7x) \\
  \ldots &
\end{align*} \]

Image Source: S. Chenney

Fourier Transforms of Important Functions

Sine and cosine transform to...

\[ \begin{align*}
  \sin(x) & \quad \Rightarrow \quad \frac{1}{2i} \\
  \cos(x) & \quad \Rightarrow \quad \frac{1}{2} \\
  \end{align*} \]
Fourier Transforms of Important Functions

- Sine and cosine transform to “frequency spikes”
- A Gaussian transforms to...
- A box filter transforms to...

Image Source: S. Chenney

Duality

- The better a function is localized in one domain, the worse it is localized in the other.
- This is true for any function

Effect of Convolution

- Convolving two functions in the image domain corresponds to taking the product of their transformed versions in the frequency domain.

\[ f \ast g \rightarrow \mathcal{F} \cdot \mathcal{G} \]

- This gives us a tool to manipulate image spectra.
  - A filter attenuates or enhances certain frequencies through this effect.

Effect of Filtering

- Noise introduces high frequencies. To remove them, we want to apply a “low-pass” filter.
- The ideal filter shape in the frequency domain would be a box. But this transfers to a spatial sinc, which has infinite spatial support.
- A compact spatial box filter transfers to a frequency sinc, which creates artifacts.
- A Gaussian has compact support in both domains. This makes it a convenient choice for a low-pass filter.
Low-Pass vs. High-Pass

Original image
Low-pass filtered
High-pass filtered

Quiz: What Effect Does This Filter Have?

Sharpening Filter

Sharpening filter
Accentuates differences with local average

Application: High Frequency Emphasis

Topics of This Lecture

- Linear Filters
  - What are they? How are they applied?
  - Application: smoothing
  - Gaussian filter
  - What does it mean to filter an image?
- Nonlinear Filters
  - Median filter
- Multi-Scale representations
  - How to properly rescale an image?
- Image derivatives
  - How to compute gradients robustly?
Non-Linear Filters: Median Filter

- **Basic idea**
  - Replace each pixel by the median of its neighbors.

- **Properties**
  - Doesn’t introduce new pixel values
  - Removes spikes: good for impulse, salt & pepper noise
  - Linear?

---

Median Filter

- The Median filter is edge preserving.

---

Median vs. Gaussian Filtering

- 3x3
- 5x5
- 7x7

---

Topics of This Lecture

- Linear filters
  - What are they? How are they applied?
  - Applications: smoothing
  - Gaussian filter
  - What does it mean to filter an image?

- Nonlinear Filters
  - Median filter

- Multi-Scale representations
  - How to properly rescale an image?

- Image derivatives
  - How to compute gradients robustly?
Image Pyramid

Low resolution

High resolution

How Should We Go About Resampling?

Let's resample the checkerboard by taking one sample at each circle.

In the top left board, the new representation is reasonable. Top right also yields a reasonable representation.

Bottom left is all black (dubious) and bottom right has checks that are too big.

Fourier Interpretation: Discrete Sampling

- Sampling in the spatial domain is like multiplying with a spike function.

- Sampling in the frequency domain is like...?

Sampling and Aliasing

Nyquist theorem:
- In order to recover a certain frequency $f$, we need to sample with at least $2f$. This corresponds to the point at which the transformed frequency spectra start to overlap.
Sampling and Aliasing

Nyquist limit

Resampling with Prior Smoothing

- Note: We cannot recover the high frequencies, but we can avoid artifacts by smoothing before resampling.

The Gaussian Pyramid

- Construction: create each level from previous one
  - Smooth and sample
- Smooth with Gaussians, in part because
  - A Gaussian*Gaussian = another Gaussian
  - $G(\sigma_1) * G(\sigma_2) = G(\sqrt{\sigma_1^2 + \sigma_2^2})$
- Gaussians are low-pass filters, so the representation is redundant once smoothing has been performed.
  - There is no need to store smoothed images at the full original resolution.

Summary: Gaussian Pyramid
**The Laplacian Pyramid**

- **Gaussian Pyramid**
  \[ G_n = G_{n-1} + \text{expand}(G_{n-1}) \]
- **Laplacian Pyramid**
  \[ L_n = G_n - \text{expand}(G_{n-1}) \]

**Why is this useful?**

**Laplacian ~ Difference of Gaussian**

**DoG = Difference of Gaussians**

Cheap approximation - no derivatives needed.

---

**Topics of This Lecture**

- Linear filters
  - What are they? How are they applied?
  - Applications: smoothing
  - Gaussian filter
  - What does it mean to filter an image?
- Nonlinear Filters
  - Median filter
- Multi-Scale representations
  - How to properly rescale an image?
- Image derivatives
  - How to compute gradients robustly?

---

**Differentiation and Convolution**

- For the 2D function \( f(x, y) \), the partial derivative is:
  \[
  \frac{\partial f(x, y)}{\partial x} = \lim_{\epsilon \to 0} \frac{f(x + \epsilon, y) - f(x, y)}{\epsilon}
  \]
- For discrete data, we can approximate this using finite differences:
  \[
  \frac{\partial f(x, y)}{\partial x} \approx \frac{f(x+1, y) - f(x, y)}{1}
  \]
- To implement the above as convolution, what would be the associated filter?
### Assorted Finite Difference Filters

- Prewitt: \( M_x = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} \); \( M_y = \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} \)
- Sobel: \( M_x = \begin{bmatrix} 2 & 0 & -2 \end{bmatrix} \); \( M_y = \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} \)
- Roberta: \( M_x = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} \); \( M_y = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \)

\[
\gg My = \text{fspecial('sobel')};
\gg outim = \text{imfilter(double(im), My)};
\gg \text{imagerc(outim)};
\gg \text{colormap gray};
\]

### Image Gradient

- The gradient of an image:
  \[
  \nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}
  \]
- The gradient points in the direction of most rapid intensity change
  \[
  \nabla f = [\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}]
  \]
- The gradient direction (orientation of edge normal) is given by:
  \[
  \theta = \tan^{-1} \left( \frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}} \right)
  \]
- The edge strength is given by the gradient magnitude
  \[
  \|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}
  \]

### Effect of Noise

- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal

### Solution: Smooth First

- Where is the edge?
- Look for peaks in \( \frac{\partial^2 (h \ast f)}{\partial x^2} \)

### Derivative Theorem of Convolution

- Differentiation property of convolution.

### Derivative of Gaussian Filter

\[
(I \ast g) \ast h = I \ast (g \ast h)
\]

Why is this preferable?
**Summary: 2D Edge Detection Filters**

- $h_x(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2 + v^2}{2\sigma^2}}$
- $\frac{\partial}{\partial x} h_x(u, v) = -\frac{u}{\sigma^2} h_x(u, v)$
- $\nabla^2 h_x(u, v) = \frac{1}{\sigma^2} \left( \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right) h_x(u, v)$

$\nabla^2$ is the Laplacian operator:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

**Note: Filters are Templates**

- Applying a filter at some point can be seen as taking a dot-product between the image and some vector.
- Filtering the image is a set of dot products.
- Insight
  - Filters look like the effects they are intended to find.
  - Filters find effects they look like.

**Derivative of Gaussian Filters**

$x$-direction

$y$-direction

**Laplacian of Gaussian (LoG)**

- Consider $\frac{\partial^2}{\partial x^2}(h * f)$

Where is the edge?  Zero-crossings of bottom graph

**Where's Waldo?**

Scene

Template

Detected template
Correlation as Template Matching

- Think of filters as a dot product of the filter vector with the image region
  - Now measure the angle between the vectors
    \[ a \cdot b = a \| b \| \cos \theta \]
    \[ \cos \theta = \frac{a \cdot b}{a \| b \|} \]
  - Angle (similarity) between vectors can be measured by normalizing length of each vector to 1.

Summary: Mask Properties

- **Smoothing**
  - Values positive
  - Sum to 1 ⇒ constant regions same as input
  - Amount of smoothing proportional to mask size
  - Remove “high-frequency” components; “low-pass” filter
- **Derivatives**
  - Opposite signs used to get high response in regions of high contrast
  - Sum to 0 ⇒ no response in constant regions
  - High absolute value at points of high contrast
- **Filters act as templates**
  - Highest response for regions that “look the most like the filter”
  - Dot product as correlation

Summary Linear Filters

- **Linear filtering:**
  - Form a new image whose pixels are a weighted sum of original pixel values
- **Properties**
  - Output is a shift-invariant function of the input (same at each image location)

Examples:

- Smoothing with a box filter
- Smoothing with a Gaussian
- Finding a derivative
- Searching for a template

Pyramid representations

- Important for describing and searching an image at all scales

References and Further Reading

- Background information on linear filters and their connection with the Fourier transform can be found in Chapters 7 and 8 of