Sampling, Filtering, Reconstruction

Image Manipulation and Computational Photography
CS294-69 Fall 2011
Maneesh Agrawala
(Some slides from James Hays, Derek Hoiem, Alexei Efros and Fredo Durand)

Sampling and Reconstruction
Sampled Representations

How to store and compute with continuous \( f(x) \)'s?

Common scheme for representation: samples

- write down the function's values at many points

Reconstruction

Making samples back into a continuous \( f(x) \)

- for output (need realizable method)
- for analysis or processing (need mathematical method)
- amounts to “guessing” what the function did in between
Sampling and Reconstruction

Simple example: a sine wave

Undersampling

What if we “missed” things between the samples?

Simple example: undersampling a sine wave

• unsurprising result: information is lost
Undersampling

What if we “missed” things between the samples?

Simple example: undersampling a sine wave

- unsurprising result: information is lost
- surprising result: indistinguishable from lower frequency

© 2006 Steve Marschner

Undersampling

What if we “missed” things between the samples?

Simple example: undersampling a sine wave

- unsurprising result: information is lost
- surprising result: indistinguishable from lower frequency
- also was always indistinguishable from higher frequencies
- aliasing: signals “traveling in disguise” as other frequencies

© 2006 Steve Marschner
Aliasing in Video

Imagine a spoked wheel moving to the right (rotating clockwise). Mark wheel with dot so we can see what’s happening.

If camera shutter is only open for a fraction of a frame time (frame time = 1/30 sec. for video, 1/24 sec. for film):

Without dot, wheel appears to be rotating slowly backwards! (counterclockwise)

Aliasing in images
What’s Happening?

Input signal:

Plot as image:

\[ x = 0:0.05:5; \text{imagesc}(\sin((2.\cdot x)^{x})) \]

Alias!
Not enough samples

Antialiasing

What can we do about aliasing?

Sample more often

- Join the Mega-Pixel craze of the photo industry
- But this can’t go on forever

Make the signal less “wiggly”

- Get rid of some high frequencies
- Will loose information
- But it looks better than aliasing
Preventing Aliasing

Introduce lowpass filters:
- remove high frequencies leaving only safe, low frequencies
- choose lowest frequency in reconstruction (disambiguate)

Linear Filtering: A Key Idea

Transformations on signals; e.g.:
- bass/treble controls on stereo
- blurring/sharpening operations in image editing
- smoothing/noise reduction in tracking

Key properties
- linearity: \( \text{filter}(f + g) = \text{filter}(f) + \text{filter}(g) \)
- shift invariance: behavior invariant to shifting the input
  - delaying an audio signal
  - sliding an image around

Can be modeled mathematically by convolution

© 2006 Steve Marschner
Moving Average

**Idea:** define new function by averaging over sliding window

A simple example to start off: smoothing

Weighted Moving Average

Can add weights to our moving average

*Weights* \([..., 0, 1, 1, 1, 1, 0, ...] / 5\)
Weighted Moving Average

bell curve (gaussian-like) weights [..., 1, 4, 6, 4, 1, ...]

Moving Average In 2D

What are the weights H?

<table>
<thead>
<tr>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ F[x, y] \]

\[ H[u, v] \]
Cross-Correlation Filtering

Equation with averaging window size \((2k+1)\times(2k+1)\):

\[
G[i, j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i + u, j + v]
\]

Allow different weights for different neighboring pixels:

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

This is called a **cross-correlation** operation and written:

\[
G = H \otimes F
\]

H is called the “filter,” “kernel,” or “mask.”

---

Gaussian Filtering

Gaussian kernel: less weight further from window center

\[
\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 90 & 90 & 90 & 90 & 0 \\
0 & 0 & 0 & 90 & 90 & 90 & 90 & 0 \\
0 & 0 & 0 & 90 & 90 & 90 & 90 & 0 \\
0 & 0 & 0 & 90 & 0 & 90 & 90 & 0 \\
0 & 0 & 0 & 90 & 90 & 90 & 90 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 90 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 90 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}
\]

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

\[
H[u, v]
\]

\[
F[x, y]
\]

\[
h(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2 + v^2}{2\sigma^2}}
\]

---

\[
\text{Slide by Steve Seitz}
\]

\[
\text{© 2006 Steve Marschner}
\]
Mean vs. Gaussian filtering

Convolution

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

A convolution operation is a cross-correlation where the filter is flipped both horizontally and vertically before being applied to the image:
\[ G[i, j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u, v] F[i - u, j - v] \]

It is written:
\[ G = H \star F \]

Suppose H is a Gaussian or mean kernel. How does convolution differ from cross-correlation?
Convolution is Nice!

Notation: \( b = c \ast a \)

Convolution is a multiplication-like operation
- commutative \( a \ast b = b \ast a \)
- associative \( a \ast (b \ast c) = (a \ast b) \ast c \)
- distributes over addition \( a \ast (b + c) = a \ast b + a \ast c \)
- scalars factor out \( \alpha a \ast b = a \ast \alpha b = \alpha (a \ast b) \)
- identity: unit impulse \( e = [\ldots, 0, 0, 1, 0, 0, \ldots] \)

\[ a \ast e = a \]

Conceptually no distinction between filter and signal

Usefulness of associativity
- often apply several filters one after another: \(((a \ast b_1) \ast b_2) \ast b_3\)
- this is equivalent to applying one filter: \(a \ast (b_1 \ast b_2 \ast b_3)\)

© 2006 Steve Marschner

Linear filtering (warm-up slide)

original

Slide credits for these examples: Bill Freeman, David Jacobs
Linear filtering (warm-up slide)

original

Filtered (no change)

Linear filtering

original
shift

original

shifted

Linear filtering

original

?
**Blurring**

- Original image
- Blurred image with filter applied

**Tricks with Convolutions**

- Convolution of `UCB` with a filter results in a blurred `UCB`
Efficient Implementation

• Both the BOX filter and the Gaussian filter are **separable** into two 1D convolutions:
  – First convolve each row with a 1D filter
  – Then convolve each column with a 1D filter.

Yucky details

What about near the image 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
  – vary filter near edge
Image Half-Sizing

This image is too big to fit on the screen. How can we reduce it?

How to generate a half-sized version?

Sub-sampling

Throw away every other row and column to create a 1/2 size image - called image sub-sampling
Sub-sampling

1/2 1/4 (2x zoom) 1/8 (4x zoom)

Aliasing! What do we do?

Slide by Steve Seitz

Downsampling

- Original image
  - Directly downsampled
Pre-filtering, Reconstruction, Resampling

Solution: filter the image, then subsample
- Filter size should double for each $\frac{1}{2}$ size reduction. Why?

Slide by Steve Seitz
Subsampling with Gaussian pre-filtering

Gaussian 1/2  G 1/4  G 1/8

Compare with...

1/2  1/4 (2x zoom)  1/8 (4x zoom)
Resampling to Enlarge

- Filter size should double for each $\frac{1}{2}$ size reduction. Why?

Gaussian (lowpass) pre-filtering

Solution: filter the image, then subsample
- Filter size should double for each $\frac{1}{2}$ size reduction. Why?

Slide by Steve Seitz
Jean Baptiste Joseph Fourier (1768-1830)

Had crazy idea (1807):

*Any* periodic function can be rewritten as a weighted sum of sines and cosines of different frequencies.

Don’t believe it?
- Neither did Lagrange, Laplace, Poisson and other big wigs
- Not translated into English until 1878!

But it’s true!
- called Fourier Series

A Sum of Sines

Our building block:

\[ A \sin(\alpha x + \phi) \]

Add enough of them to get any signal \( f(x) \) you want!

How many degrees of freedom?

What does each control?

Which one encodes the coarse vs. fine structure of the signal?

\[ f(\text{target}) = f_1 + f_2 + f_3 + \ldots + f_n \]
For every $\omega$ from 0 to inf, $F(\omega)$ holds the amplitude $A$ and phase $\phi$ of the corresponding sine $A \sin(\omega x + \phi)$.

How can $F$ hold both? Complex number trick!

$F(\omega) = R(\omega) + iI(\omega)$

$A = \pm \sqrt{R(\omega)^2 + I(\omega)^2}$  \hspace{1cm} $\phi = \tan^{-1} \frac{I(\omega)}{R(\omega)}$

We can always go back:

$F(\omega) \rightarrow$ Inverse Fourier Transform $\rightarrow f(x)$

Time and Frequency

example: $g(t) = \sin(2pf \ t) + (1/3)\sin(2p(3f) \ t)$
**Time and Frequency**

example: \( g(t) = \sin(2\pi f t) + \frac{1}{3}\sin(2\pi (3f) t) \)

**Frequency Spectra**

example: \( g(t) = \sin(2\pi f t) + \frac{1}{3}\sin(2\pi (3f) t) \)
Frequency Spectra

Usually, frequency is more interesting than the phase
Frequency Spectra

Frequency Spectra
Frequency Spectra

\[ A \sum_{k=1}^{\infty} \frac{1}{k} \sin(2\pi kt) \]
Extension to 2D

in Matlab, check out: imagesc(log(abs(fftshift(fft2(im)))));

Man-made Scene
Can Change Spectrum, then Reconstruct

Low and High Pass filtering
The Convolution Theorem

The greatest thing since sliced (banana) bread!

The Fourier transform of the convolution of two functions is the product of their Fourier transforms

\[ F[g * h] = F[g]F[h] \]

The inverse Fourier transform of the product of two Fourier transforms is the convolution of the two inverse Fourier transforms

\[ F^{-1}[gh] = F^{-1}[g] * F^{-1}[h] \]

Convolution in spatial domain is equivalent to multiplication in frequency domain!

2D Convolution Theorem Example

\[ f(x,y) * h(x,y) = g(x,y) \]

\[ |F(s_x,s_y)| * |H(s_x,s_y)| = |G(s_x,s_y)| \]
Fourier Transform Pairs

Spatial domain

Low-pass:
- box(x)

Band-pass:
- gauss(x; σ)

High-pass:
- sinc(s)

Frequency domain

Low-pass:
- sinc(s)

Band-pass:
- sinc(s)

High-pass:
- box(x)

Low-pass, Band-pass, High-pass filters

Low-pass:

High-pass / band-pass:
Edges in Images

What Does Blurring Remove?

original
What Does Blurring Remove?

smoothed (5x5 Gaussian)

High-Pass Filter

smoothed – original
Unsharp Masking

- =

+ α =

Hybrid Images

[Oliva, Torralba & Schyns 05]

Figure 2: hybrid images are generated by superimposing two images at two different spatial scales: the low-spatial scale is obtained by filtering one image with a low-pass filter, and the high-spatial scale is obtained by filtering a second image with a high-pass filter. The final hybrid image is composed by adding these two filtered images.

Assignment 0: Implement Hybrid Images
Hybrid Images

Image Pyramids

Idea: Represent N×N image as a "pyramid" of 1×1, 2×2, 4×4, ..., 2^k×2^k images (assuming N=2^k)

Known as a Gaussian Pyramid [Burt and Adelson, 1983]
- In computer graphics, a mip map [Williams, 1983]
- A precursor to wavelet transform
Gaussian Pyramid Construction

Repeat
• Filter
• Subsample

Until minimum resolution reached
• can specify desired number of levels (e.g., 3-level pyramid)

The whole pyramid is only 4/3 the size of the original image!
Band-pass filtering

Gaussian Pyramid (low-pass images)

Laplacian Pyramid

How can we reconstruct (collapse) this pyramid into the original image?
Pyramid Based Graphics [Ogden, Adelson, Bergen & Burt 84]

Laplacian pyramid introduced in classic paper: [Burt & Adelson 83]

Applications summarized in 84 paper

- Image interpolation
- Merging images into mosaics
- Creating realistic shadows and shading
- Generation of natural looking textures using fractals
- Real-time animation of fractals

Assignment 0: Extra Credit

---

Image Gradient

The gradient of an image:
\[ \nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right] \]

The gradient points in the direction of most rapid change in intensity

\[ \nabla f = \left[ \frac{\partial f}{\partial x}, 0 \right] \quad \nabla f = \left[ 0, \frac{\partial f}{\partial y} \right] \]

The gradient direction is given by:
\[ \theta = \tan^{-1} \left( \frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}} \right) \]
- how does this relate to the direction of the edge?

The edge strength is given by the gradient magnitude
\[ \| \nabla f \| = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2} \]
Effects of Noise

Consider a single row or column of the image

- Plotting intensity as a function of position gives a signal

How to compute a derivative?

Where is the edge?

Solution: Smooth First

Where is the edge?  Look for peaks in $\frac{\partial}{\partial x} (h \ast f)$
Derivative Theorem of Convolution

\[ \frac{\partial}{\partial x} (h \ast f) = (\frac{\partial}{\partial x} h) \ast f \]

This saves us one operation:

![Graphs showing convolution and its derivative]

Laplacian of Gaussian

Consider \( \frac{\partial^2}{\partial x^2} (h \ast f) \)

![Graphs showing second derivative and its convolution]

Where is the edge? Zero-crossings of bottom graph
2D Edge Detection Filters

Gaussian

\[
h_\sigma(u, v) = \frac{1}{2\pi \sigma^2} e^{-\frac{u^2 + v^2}{2\sigma^2}}
\]

is the Laplacian operator:

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

derivative of Gaussian

\[
\frac{\partial}{\partial x} h_\sigma(u, v)
\]

Laplacian of Gaussian

\[
\nabla^2 h_\sigma(u, v)
\]