Review – Frequency Domain Techniques

• The name “filter” is borrowed from frequency domain processing
• Accept or reject certain frequency components
• Fourier (1807): Periodic functions could be represented as a weighted sum of sines and cosines

Review – The Discrete Fourier Transform

• Forward transform

\[ F[m,n] = \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} f[k,l] e^{-\frac{2\pi i}{MN} (km + ln)} \]

• Inverse transform

\[ f[k,l] = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} F[m,n] e^{\frac{2\pi i}{MN} (km + ln)} \]
**Review - The Discrete Fourier transform**

- **Horizontal orientation**
- **Vertical orientation**
- **45 deg.**
- **Log power spectrum**

**Review - The Convolusion Theorem**

- 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!

**Review - Filtering in frequency domain**

- **FFT**
- **Inverse FFT**

**Review - Low-pass, Band-pass, High-pass filters**

- **low-pass:**
- **High-pass / band-pass:**
**Template matching**

- Goal: find \( \mathbf{f} \) in image

- Main challenge: What is a good similarity or distance measure between two patches?
  - Correlation
  - Zero-mean correlation
  - Sum Square Difference
  - Normalized Cross Correlation

**Matching with filters**

- Goal: find \( \mathbf{f} \) in image

- Method 0: filter the image with an eye patch

\[
h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l]
\]

- Method 1: filter the image with zero-mean eye

\[
h[m,n] = \sum_{k,l} (f[k,l] - \bar{f})(g[m+k,n+l])
\]

- Method 2: SSD

\[
h[m,n] = \sum_{k,l} (g[k,l] - f[m+k,n+l])^2
\]

**What went wrong?**

Response is stronger for higher intensity

**True detections**

**False detections**
Matching with filters

- Goal: find an object in the image
- Method 2: SSD

$$h[m,n] = \sum_{k,l}(g[k,l] - f[m+k,n+l])^2$$

What's the potential downside of SSD?

SSD sensitive to average intensity

Input

I - sqrt(SSD)

Slide: Hoiem

Matching with filters

- Goal: find an object in the image
- Method 3: Normalized cross-correlation

$$h[m,n] = \frac{\sum_{k,l}(g[k,l] - \bar{g})(f[m-k,n-l] - \bar{f}_{m,n})}{\left(\sum_{k,l}(g[k,l] - \bar{g})^2\sum_{k,l}(f[m-k,n-l] - \bar{f}_{m,n})^2\right)^{0.5}}$$

Matlab: `normxcorr2(template, im)`

Input

Normalized X-Correlation

Thresholded Image

Slide: Hoiem

Matching with filters

- Goal: find an object in the image
- Method 3: Normalized cross-correlation

Input

Normalized X-Correlation

Thresholded Image

Slide: Hoiem
**Q: What is the best method to use?**

A: Depends

- SSD: faster, sensitive to overall intensity
- Normalized cross-correlation: slower, invariant to local average intensity and contrast

**Q: What if we want to find larger or smaller eyes?**

A: Image Pyramid

---

**Image pyramids**

Image information occurs at all spatial scales

- Gaussian pyramid
- Laplacian pyramid
- Wavelet/QMF pyramid
- Steerable pyramid

---

Slide credit: B. Freeman and A. Torralba
Image pyramids

Image information occurs at all spatial scales

• Gaussian pyramid
• Laplacian pyramid
• Wavelet/QMF pyramid
• Steerable pyramid

The Gaussian pyramid

• Smooth with Gaussians, because
  – A Gaussian * Gaussian = another Gaussian
• Gaussians are low pass filters, so representation is redundant.
• Gaussian pyramid creates versions of the input image at multiple resolutions.
• This is useful for analysis across different spatial scales, but doesn’t separate the image into different frequency bands.

The computational advantage of pyramids

Fig 1. A one-dimensional graphic representation of the process which generates a Gaussian pyramid. Each row of dots represents nodes within a level of the pyramid. The value of each node in the zero level is just the gray level of a corresponding image pixel. The value of each node in a high level is the weighted average of node values in the next lower level. Note that node spacing doubles from level to level, while the same weighting pattern or “generating kernel” is used to generate all levels.

[Burt and Adelson, 1983]
The Gaussian Pyramid

Convolution and subsampling as a matrix multiply (1D case)

\[ x_2 = G_1 x_1 \]

\[ G_1 = \]

\[
\begin{array}{cccccccccccccccc}
1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 \end{array} \]

\( (\text{Normalization constant of } 1/16 \text{ omitted for visual clarity.}) \)

Next pyramid level

\[ x_3 = G_2 x_2 \]

\[ G_2 = \]

\[
\begin{array}{cccccccccccc}
1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 & 4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \]
The combined effect of the two pyramid levels

\[ x_3 = G_2 G_1 x_1 \]

\[ G_2 G_1 = \]

\[
\begin{array}{ccccccccccccccccccccccc}
1 & 4 & 10 & 20 & 31 & 40 & 44 & 40 & 31 & 20 & 10 & 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 4 & 10 & 20 & 31 & 40 & 44 & 40 & 31 & 20 & 10 & 4 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 10 & 20 & 31 & 40 & 44 & 40 & 30 & 16 & 4 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 & 10 & 20 & 25 & 16 & 4 & 0 & 0 & 0 \\
\end{array}
\]

Gaussian pyramids used for

- up- or down- sampling images.
- Multi-resolution image analysis
  - Look for an object over various spatial scales
  - Coarse-to-fine image processing: form blur estimate or the motion analysis on very low-resolution image, upsample and repeat. Often a successful strategy for avoiding local minima in complicated estimation tasks.
Template Matching with Image Pyramids

Input: Image, Template
1. Match template at current scale
2. Downsample image
3. Repeat 1-2 until image is very small
4. Take responses above some threshold, perhaps with non-maxima suppression

Coarse-to-fine Image Registration
1. Compute Gaussian pyramid
2. Align with coarse pyramid
3. Successively align with finer pyramids
   - Search smaller range

Why is this faster?

Are we guaranteed to get the same result?

Image pyramids
Image information occurs at all spatial scales

- Gaussian pyramid
- Laplacian pyramid
- Wavelet/QMF pyramid
- Steerable pyramid

The Laplacian Pyramid
- Synthesis
  - Compute the difference between upsampled Gaussian pyramid level and Gaussian pyramid level.
  - Band pass filter - each level represents spatial frequencies (largely) unrepresented at other level.

- Laplacian pyramid provides an extra level of analysis as compared to Gaussian pyramid by breaking the image into different isotropic spatial frequency bands.
The Laplacian Pyramid

\[ x_1 \rightarrow G_x x_1 \rightarrow \ldots \]

\[ F_i G_i x_1 \]

\[ (I - F_i G_i) x_1 \]

Slide credit: B. Freeman and A. Torralba
The Laplacian Pyramid

The Laplacian Pyramid

The Laplacian Pyramid

Upsampling

\[ y_2 = F_3 x_3 \]

Insert zeros between pixels, then apply a low-pass filter, \([1 \text{ 4 6 4 1}]\)

\[
F_3 = \begin{bmatrix}
6 & 1 & 0 & 0 \\
4 & 4 & 0 & 0 \\
1 & 6 & 1 & 0 \\
0 & 4 & 4 & 0 \\
0 & 1 & 6 & 1 \\
0 & 0 & 4 & 4 \\
0 & 0 & 1 & 6 \\
0 & 0 & 0 & 4
\end{bmatrix}
\]
Showing, at full resolution, the information captured at each level of a Gaussian (top) and Laplacian (bottom) pyramid.

Laplacian pyramid reconstruction algorithm: recover $x_1$ from $L_1$, $L_2$, $L_3$ and $x_4$

$G#$ is the blur-and-downsample operator at pyramid level #
$F#$ is the blur-and-upsample operator at pyramid level #

Laplacian pyramid elements:

- $L_1 = (I - F_1 G_1) x_1$
- $L_2 = (I - F_2 G_2) x_2$
- $L_3 = (I - F_3 G_3) x_3$
- $x_2 = G_1 x_1$
- $x_3 = G_2 x_2$
- $x_4 = G_3 x_3$

Reconstruction of original image ($x_1$) from Laplacian pyramid elements:

- $x_3 = L_3 + F_3 x_4$
- $x_2 = L_2 + F_2 x_3$
- $x_1 = L_1 + F_1 x_2$
Laplacian pyramid applications

- Texture synthesis
- Image compression
- Noise removal

The Laplacian Pyramid as a Compact Image Code

PETER J. BURT, MEMBER, IEEE, AND EDWARD H. ADelson
**Image blending**

- Build Laplacian pyramid for both images: LA, LB
- Build Gaussian pyramid for mask: G
- Build a combined Laplacian pyramid:
  \[ L(j) = G(j) LA(j) + (1-G(j)) LB(j) \]
- Collapse L to obtain the blended image

---

**Image pyramids**

Image information occurs at all spatial scales

- Gaussian pyramid
- Laplacian pyramid
- Wavelet/QMF pyramid
- Steerable pyramid

---

**Wavelet/QMF pyramid**

- Subband coding
  - Wavelet or QMF (quadrature mirror filter) pyramid provides some splitting of the spatial frequency bands according to orientation (although in a somewhat limited way).
  - Image is decomposed into a set of band-limited components (subbands).
  - Original image can be reconstructed without error by reassembling these subbands.
2D Haar transform

Basic elements:

\[
\begin{pmatrix}
1 & 1 \\
1 & -1 
\end{pmatrix}
\]

Low pass

\[
\begin{pmatrix}
1 & 1 \\
1 & -1 
\end{pmatrix} = \mathcal{F}
\]

High pass vertical

\[
\begin{pmatrix}
1 & 1 \\
-1 & 1 
\end{pmatrix} = \mathcal{F}
\]

High pass horizontal

\[
\begin{pmatrix}
1 & -1 \\
1 & 1 
\end{pmatrix} = \mathcal{F}
\]

High pass diagonal

\[
\begin{pmatrix}
1 & -1 \\
-1 & 1 
\end{pmatrix} = \mathcal{F}
\]

Pyramid cascade

Wavelet/QMF representation

Figure 4.12: Idealized diagram of the partition of the frequency plane resulting from a 3-level pyramid cascade of separable 2-band filters. The top plot represents the frequency spectrum of the original image, with axes ranging from $-\pi$ to $\pi$. This is divided into four subbands at the next level. On each subsequent level, the lowpass subband of each is built in sublatticed form.

**Image pyramids**

Image information occurs at all spatial scales

- Gaussian pyramid
- Laplacian pyramid
- Wavelet/QMF pyramid
- Steerable pyramid

---

**Steerable Pyramid**

We may combine steerability with pyramids to get a Steerable Laplacian Pyramid as shown below.

Images from: [http://www.cis.upenn.edu/~eero/steerpyr.html](http://www.cis.upenn.edu/~eero/steerpyr.html)  
Slide credit: B. Freeman and A. Torralba
Steerable Pyramid

But we need to get rid of the corner regions before starting the recursive circular filtering.

Image pyramids

- Gaussian
- Laplacian
- Wavelet/QMF
- Steerable pyramid

There is also a high pass residual...


Image pyramids

- Gaussian
- Laplacian
- Wavelet/QMF
- Steerable pyramid

Progressively blurred and subsampled versions of the image. Adds scale invariance to fixed-size algorithms.

Slide credit: B. Freeman and A. Torralba.
**Image pyramids**

- **Gaussian**
  - Shows the information added in Gaussian pyramid at each spatial scale. Useful for noise reduction & coding.
  - Progressively blurred and subsampled versions of the image. Adds scale invariance to fixed-size algorithms.

- **Laplacian**
  - Shows the information added in Gaussian pyramid at each spatial scale. Useful for noise reduction & coding.

- **Wavelet/QMF**
  - Bandpassed representation, complete, but with aliasing and some non-oriented subbands.

- **Steerable pyramid**
  - Shows components at each scale and orientation separately. Non-aliased subbands. Good for texture and feature analysis. But overcomplete and with HF residual.

---

**Schematic pictures of each matrix transform**

Shown for 1-d images

The matrices for 2-d images are the same idea, but more complicated, to account for vertical, as well as horizontal, neighbor relationships.

\[
\vec{F} = \mathbf{U} \vec{f}
\]

- **Vectorized image**
- **Fourier transform, or Wavelet transform, or Steerable pyramid transform**
Fourier transform

\[ \text{Fourier transform} \]

Fourier bases are global: each transform coefficient depends on all pixel locations.

Gaussian pyramid

\[ \text{Gaussian pyramid} \]

Overcomplete representation. Low-pass filters, sampled appropriately for their blur.

Laplacian pyramid

\[ \text{Laplacian pyramid} \]

Overcomplete representation. Transformed pixels represent bandpassed image information.

Wavelet (QMF) transform

\[ \text{Wavelet (QMF) transform} \]

Ortho-normal transform (like Fourier transform), but with localized basis functions.
**Why use image pyramids?**

- Handle real-world size variations with a constant-size vision algorithm.
- Remove noise
- Analyze texture
- Recognize objects
- Label image features
- Image priors can be specified naturally in terms of wavelet pyramids.

---

**Reading Assignment #3 – Hybrid Images**

- Due on 20th of December