Visual Computing: Pyramids and Wavelets

Prof. Marc Pollefeys
Prof. Markus Gross
Last lecture

- **PCA** (or KL transform)

\[
\hat{I} = I - \bar{I} = \begin{pmatrix} \ldots \\ U_k \\ \ldots \end{pmatrix} \approx \begin{pmatrix} \ldots \\ U_k \end{pmatrix} \begin{pmatrix} \Sigma_k \end{pmatrix} \begin{pmatrix} V_k^T \end{pmatrix} = \begin{pmatrix} \ldots \\ \mathbf{c}_i \end{pmatrix} = \Sigma_k V_k^T
\]

- Strongly correlated samples, equal energies
- After KLT: uncorrelated samples, most of the energy in first coefficient
Eigenspace matching

- Consider PCA (aka KLT)

\[ \hat{I}_i \approx \hat{E} p_i \]

\[ \approx \begin{bmatrix} U \cdot \Sigma_k \cdot V_k^T \end{bmatrix} \]

- Then,

\[ I_i - I = \hat{I}_i - \bar{I} \approx E (p_i - p) \]

\[ \|I_i - I\| \approx \|p_i - p\| \]

Closest rank-k approximation property of SVD

approximate \( \arg \min_{i} D_i = \|I_i - I\| \)

with \( \arg \min \|p_i - p\| \)

Much cheaper to compute!

\[ E^3 \]

average face

+ \( c_1 \) + \( c_2 \) + \( c_3 \) + \( c_4 \)
Eigenimages for recognition (cont.)

[Ruiz-del-Solar and Navarrete, 2005]
Limitations of Eigenfaces

Differences due to varying illumination can be much larger than differences between faces!

[Belhumeur, Hespanha, Kriegman, 1997]
KEY IDEAS:
- Find directions where ratio of between:within individual variance are maximized
- Linearly project to basis where dimension with good signal:noise ratio are maximized
Fisher linear discriminant analysis

- Eigenimage method maximizes “scatter” within the linear subspace over the entire image set – regardless of classification task
  \[ W_{opt} = \arg \max_W \left( \det \left( W R W^H \right) \right) \]

- Fisher linear discriminant analysis (1936): maximize between-class scatter, while minimizing within-class scatter

\[ W_{opt} = \arg \max_W \left( \frac{\det \left( W R_B W^H \right)}{\det \left( W R_W W^H \right)} \right) \]

\[ R_B = \sum_{i=1}^{c} N_i \left( \mu_i - \mu \right) \left( \mu_i - \mu \right)^H \]

\[ R_W = \sum_{i=1}^{c} \sum_{\Gamma_l \in \text{Class}(i)} \left( \Gamma_l - \mu_i \right) \left( \Gamma_l - \mu_i \right)^H \]
Fisher linear discriminant analysis (cont.)

- Solution: Generalized eigenvectors $\overline{w}_i$ corresponding to the $K$ largest eigenvalues $\{\lambda_i \mid i = 1, 2, \ldots, K\}$, i.e.

\[
R_B \overline{w}_i = \lambda_i R_W \overline{w}_i, \quad i = 1, 2, \ldots, K
\]

- Problem: within-class scatter matrix $R_w$ at most of rank $L-c$, hence usually singular.

- Apply KLT first to reduce dimension of feature space to $L-c$ (or less), proceed with Fisher LDA in low-dimensional space
Eigenfaces vs. Fisherfaces

2d example:
Samples for 2 classes are projected onto 1d subspace using the KLT (aka PCA) or Fisher LDA (FLD). PCA preserves maximum energy, but the 2 classes are no longer distinguishable. FLD separates the classes by choosing a better 1d subspace.

[Belhumeur, Hespanha, Kriegman, 1997]
Eigenfaces vs. Fisherfaces

Differences due to varying illumination can be much larger than differences between faces!

[Belhumeur, Hespanha, Kriegman, 1997]
Fisher images and varying illumination

- All images of same Lambertian surface with different illumination (without shadows) lie in a 3d linear subspace
- Single point source at infinity

\[ f(x, y) = a(x, y)(l^T n(x, y))L \]

- Superposition of arbitrary number of point sources at infinity still in same 3d linear subspace, due to linear superposition of each contribution to image
- Fisher images can eliminate within-class scatter
Face recognition with Eigenfaces and Fisherfaces

![Graph showing identification rate vs. number of features for FERET data base, 254 classes, 3 images per class. The graph compares EIGENFACES and FISHERFACES methods.]
Fisher images trained to recognize glasses

<table>
<thead>
<tr>
<th>Glasses Recognition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>Eigenface</td>
</tr>
<tr>
<td>Fisherface</td>
</tr>
</tbody>
</table>

[Belhumeur, Hespanha, Kriegman, 1997]
Appearance manifold approach

- for every object
  - sample the set of viewing conditions
- use these images as feature vectors
- apply a PCA over all the images
- keep the dominant PCs
- sequence of views for 1 object represent a manifold in space of projections
- what is the nearest manifold for a given view?

[Nayar et al. ’96]
Object-pose manifold

- Appearance changes projected on PCs (1D pose changes)
- Sufficient characterization for recognition and pose estimation

[Nayar et al. ’96]
Real-time recognition system

[Nayar et al. ’96]
JPEG image compression

Lenna, 256x256 RGB
Baseline JPEG: 4572 bytes
Campbell-Robson contrast sensitivity curve

We don’t resolve high frequencies too well…
… let’s use this to compress images… JPEG!
Lossy Image Compression (JPEG)

Block-based Discrete Cosine Transform (DCT)
JPEG Encoding and Decoding

Encoding

Decoding

DC Huffman

IDPCM

DC

Q

DCT

8x8 block

Quantization Matrix

DC

IDCT

AC Huffman

Q⁻¹

AC

8x8 block

www.jpeg.org
Using DCT in JPEG

A variant of discrete Fourier transform
  - Real numbers
  - Fast implementation

Block size
  - small block
    - faster
    - correlation exists between neighboring pixels
  - large block
    - better compression in smooth regions
Using DCT in JPEG

The first coefficient \( B(0,0) \) is the DC component, the average intensity.
The top-left coeffs represent low frequencies, the bottom right – high frequencies.
Image compression using DCT

DCT enables image compression by concentrating most image information in the low frequencies.

Loose unimportant image info (high frequencies) by cutting $B(u,v)$ at bottom right.

The decoder computes the inverse DCT – IDCT.

Quantization Table:

<table>
<thead>
<tr>
<th>3</th>
<th>5</th>
<th>7</th>
<th>9</th>
<th>11</th>
<th>13</th>
<th>15</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>13</td>
<td>15</td>
<td>17</td>
<td>19</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>11</td>
<td>13</td>
<td>15</td>
<td>17</td>
<td>19</td>
<td>21</td>
</tr>
<tr>
<td>9</td>
<td>11</td>
<td>13</td>
<td>15</td>
<td>17</td>
<td>19</td>
<td>21</td>
<td>23</td>
</tr>
<tr>
<td>11</td>
<td>13</td>
<td>15</td>
<td>17</td>
<td>19</td>
<td>21</td>
<td>23</td>
<td>25</td>
</tr>
<tr>
<td>13</td>
<td>15</td>
<td>17</td>
<td>19</td>
<td>21</td>
<td>23</td>
<td>25</td>
<td>27</td>
</tr>
<tr>
<td>15</td>
<td>17</td>
<td>19</td>
<td>21</td>
<td>23</td>
<td>25</td>
<td>27</td>
<td>29</td>
</tr>
<tr>
<td>17</td>
<td>19</td>
<td>21</td>
<td>23</td>
<td>25</td>
<td>27</td>
<td>29</td>
<td>31</td>
</tr>
</tbody>
</table>
Entropy Coding (Huffman code)

- The code words, if regarded as a binary fractions, are pointers to the particular interval being coded.
- In Huffman code, the code words point to the base of each interval.
- The average code word length is $H = -\sum p(s) \log_2 p(s)$ -> optimal
JPEG compression comparison
JPEG image compression

Lenna, 256x256 RGB
Baseline JPEG: 4572 bytes
Scale-space representations

- From an original signal $f(x)$ generate a parametric family of signals $f^t(x)$ where fine-scale information is successively suppressed.

$$f(x) = f_0(x) \quad \Downarrow \quad f^t(x)$$

- Family of signals generated by successive smoothing with a Gaussian filter.
- Zero-crossings of 2\textsuperscript{nd} derivative: Fewer features at coarser scales.

[Witkin 1983]
Image pyramid
Applications of scaled representations

• Search for correspondence
  – look at coarse scales, then refine with finer scales

• Edge tracking
  – a “good” edge at a fine scale has parents at a coarser scale

• Control of detail and computational cost in matching
  – e.g. finding stripes
  – terribly important in texture representation
Example: CMU face detection
The Gaussian pyramid

• Smooth with gaussians, because
  – a gaussian*gaussian=another gaussian

• Synthesis
  – smooth and sample

• Analysis
  – take the top image

• Gaussians are low pass filters, so representation is redundant
GAUSSIAN PYRAMID

$g_0 = \text{IMAGE}$

$g_L = \text{REDUCE}[g_{L-1}]$
The Laplacian Pyramid

• Synthesis
  – preserve difference between upsampled Gaussian pyramid level and Gaussian pyramid level
  – band pass filter - each level represents spatial frequencies (largely) unrepresented at other levels

• Analysis
  – reconstruct Gaussian pyramid, take top layer
LoG vs. DoG

Laplacian of Gaussian

Difference of Gaussians
1D Discrete Wavelet Transform

- Recursive application of a two-band filter bank to the lowpass band of the previous stage yields octave band splitting:
Haar Transform

- Haar transform $H$
  - Sample $h_k(x)$ at $\{m/N\}$
    - $m = 0, ..., N-1$
  - Real and orthogonal
  - Transition at each scale $p$ is localized according to $q$

- Basis images of 2-D (separable) Haar transform
  - Outer product of two basis vectors
Compare Basis Images of DCT and Haar

See also: Jain’s Fig.5.2 pp136
Summary on Haar Transform

- Two major sub-operations
  - Scaling captures info. at different frequencies
  - Translation captures info. at different locations
- Can be represented by filtering and downsampling
- Relatively poor energy compaction
Cascade analysis/synthesis filterbanks
Successive Wavelet/Subband Decomposition

Successive lowpass/highpass filtering and downsampling

- on different level: capture transitions of different frequency bands
- on the same level: capture transitions at different locations

Figure from Matlab Wavelet Toolbox Documentation
two-filterbanks with perfect reconstruction

- Impulse responses, analysis filters:
  - **Lowpass**
  \[
  \begin{pmatrix}
  -\frac{1}{4} & \frac{1}{2} & \frac{3}{2} & \frac{1}{4}
  \end{pmatrix}
  \]
  - **highpass**
  \[
  \begin{pmatrix}
  \frac{1}{4} & -\frac{1}{2} & \frac{1}{4}
  \end{pmatrix}
  \]

- Impulse responses, synthesis filters
  - **Lowpass**
  \[
  \begin{pmatrix}
  \frac{1}{4} & \frac{1}{2} & \frac{1}{4}
  \end{pmatrix}
  \]
  - **highpass**
  \[
  \begin{pmatrix}
  \frac{1}{4} & \frac{1}{2} & -\frac{3}{2} & \frac{1}{4}
  \end{pmatrix}
  \]

- Mandatory in JPEG2000
- Frequency responses:

  - “Biorthogonal 5/3 filters”
  - “LeGall filters”

**Graph**

- Frequency response
  - $|h_0|$ and $|h_1|$
Lifting

- Analysis filters

\[ x[2n] \rightarrow \lambda_1 \rightarrow \lambda_2 \rightarrow \ldots \rightarrow \lambda_{L-1} \rightarrow \lambda_L \rightarrow y_0 \]

\[ x[2n+1] \rightarrow \sum \rightarrow \lambda_{L-1} \rightarrow \lambda_L \rightarrow y_1 \]

- \( L \) “lifting steps”

- First step can be interpreted as prediction of odd samples from the even samples

[Sweldens 1996]
Lifting (cont.)

- Synthesis filters

  Even samples $x[2n]$

- Perfect reconstruction (biorthogonality) is directly build into lifting structure

- Powerful for both implementation and filter/wavelet design
Example: Lifting implementation of 5/3 filter

Verify by considering response to unit impulse in even and odd input channel.
Operation flow of JPEG2000

Figure 2 Operation flow of the JPEG 2000 standard.
JPEG vs. JPEG2000

Lenna, 256x256 RGB
Baseline JPEG: 4572 bytes

Lenna, 256x256 RGB
JPEG-2000: 4572 bytes
Examples

JPEG2K

VS.

JPEG

Fig. 20. Reconstructed images compressed at 0.125 bpp by means of (a) JPEG and (b) JPEG2000

Fig. 21. Reconstructed images compressed at 0.25 bpp by means of (a) JPEG and (b) JPEG2000

From Christopoulos
(IEEE Trans. on CE 11/00)