CSEP 590
Data Compression
Autumn 2007

Lossy Image Compression
Transform Coding
JPEG

Lossy Image Compression Methods
• DCT Compression
  – JPEG
• Scalar quantization (SQ).
• Vector quantization (VQ).
• Wavelet Compression
  – SPIHT
  – UWIC (University of Washington Image Coder)
  – EBCOT
  – JPEG 2000

JPEG Standard
• JPEG - Joint Photographic Experts Group
• JPEG 2000 uses to wavelet compression.

Barbara
original
JPEG
32:1 compression ratio
.25 bits/pixel (8 bits)
VQ
Wavelet-SPIHT

JPEG
VQ
Images and the Eye

- Images are meant to be viewed by the human eye (usually).
- The eye is very good at "interpolation", that is, the eye can tolerate some distortion. So lossy compression is not necessarily bad. The eye has more acuity for luminance (gray scale) than chrominance (color).
  - Gray scale is more important than color.
  - Compression is usually done in the YUV color coordinates, Y for luminance and U, V for color.
  - U and V should be compressed more than Y
  - This is why we will concentrate on compressing gray scale (8 bits per pixel) images.

Distortion

- Lossy compression: \( x \neq \hat{x} \)
- Measure of distortion is commonly mean squared error (MSE). Assume \( x \) has \( n \) real components (pixels).
  \[
  MSE = \frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{x}_i)^2
  \]

PSNR

- Peak Signal to Noise Ratio (PSNR) is the standard way to measure fidelity.
  \[
  PSNR = 10 \log_{10} \left( \frac{m^2}{MSE} \right)
  \]
  where \( m \) is the maximum value of a pixel possible.
  For gray scale images (8 bits per pixel) \( m = 255 \).

- PSNR is measured in decibels (dB).
  - .5 to 1 dB is said to be a perceptible difference.
  - Decent images start at about 30 dB

Rate-Fidelity Curve

- Increasing
- Slope decreasing
PSNR is not Everything

PSNR = 25.8 dB

VQ

PSNR Reflects Fidelity (1)

PSNR = 25.8

0.63 bpp

12.8 : 1

VQ

PSNR Reflects Fidelity (2)

PSNR = 24.2

0.31 bpp

25.6 : 1

VQ

PSNR Reflects Fidelity (3)

PSNR = 23.2

0.16 bpp

51.2 : 1

VQ

Idea of Transform Coding

• Transform the input pixels \(x_0, x_2, ..., x_{N-1}\) into coefficients \(c_0, c_1, ..., c_{N-1}\) (real values)
  – The coefficients have the property that most of them are near zero
  – Most of the “energy” is compacted into a few coefficients
• Quantize the coefficients
  – This is where there is loss, since coefficients are only approximated
  – Important coefficients are kept at higher precision
• Entropy encode the quantization symbols

Decoding

• Entropy decode the quantized symbols
• Compute approximate coefficients \(c'_0, c'_1, ..., c'_{N-1}\) from the quantized symbols.
• Inverse transform \(c'_0, c'_1, ..., c'_{N-1}\) to \(x'_0, x'_1, ..., x'_{N-1}\) which is a good approximation of the original \(x_0, x_2, ..., x_{N-1}\).
**Block Diagram of Transform Coding**

Encoder

Input $x$

```
<table>
<thead>
<tr>
<th>transform</th>
<th>quantization</th>
<th>allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Entropy coding

Output $s$

---

Decoder

```
<table>
<thead>
<tr>
<th>inverse transform</th>
<th>decode symbols</th>
<th>entropy decoding</th>
</tr>
</thead>
<tbody>
<tr>
<td>s' $x'$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

**Mathematical Properties of Transforms**

- **Linear Transformation** - Defined by a real nxn matrix $A = (a_{ij})$
  
  \[
  \begin{bmatrix}
  a_{00} & \cdots & a_{0,N-1} \\
  \vdots & \ddots & \vdots \\
  a_{N-1,0} & \cdots & a_{N-1,N-1}
  \end{bmatrix}
  \begin{bmatrix}
  x_0 \\
  \vdots \\
  x_{N-1}
  \end{bmatrix} =
  \begin{bmatrix}
  c_0 \\
  \vdots \\
  c_{N-1}
  \end{bmatrix}
  \]

- **Orthonormality** $A^{-1} = A^T$ (transpose)

**Why Coefficients**

\[
A^T c = x
\]

\[
\begin{bmatrix}
  a_{00} & \cdots & a_{0,N-1} \\
  \vdots & \ddots & \vdots \\
  a_{N-1,0} & \cdots & a_{N-1,N-1}
  \end{bmatrix}
  \begin{bmatrix}
  c_0 \\
  \vdots \\
  c_{N-1}
  \end{bmatrix} =
  \begin{bmatrix}
  x_0 \\
  \vdots \\
  x_{N-1}
  \end{bmatrix}
  \]

**Why Orthonormality**

- The energy of the data equals the energy of the coefficients

  \[
  \sum_{i=0}^{N-1} c_i^2 = c^T c = (A x)^T (A x) = x^T A^T A x = x^T x = \sum_{i=0}^{N-1} x_i^2
  \]

**Squared Error is Preserved with Orthonormal Transformations**

- In lossy coding we only send an approximation $c'_i$ of $c_i$ because it takes fewer bits to transmit the approximation.

  \[
  \sum_{i=0}^{N-1} x_i^2 = \sum_{i=0}^{N-1} (c_i - c'_i)^2 = (c - c')^T (c - c') = (A x - A' x')^T (A x - A' x')
  \]

  \[
  = (A(x - x'))^T (A(x - x')) = ((x - x')^T A^T) (A(x - x'))
  \]

  \[
  = (x - x')^T (A^T A) (x - x') = (x - x')^T (x - x')
  \]

  \[
  = \sum_{i=0}^{N-1} (x_i - x'_i)^2
  \]

  Squared error in original.

**Compaction Example**

\[
A = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}
\]

\[
A^2 = A \Rightarrow A^{-1} = A
\]

\[
A^{-1} = A^T = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}
\]

Orthonormal

\[
A = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}
\]

\[
\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} b \\ 0 \end{bmatrix} = \begin{bmatrix} \sqrt{2}b \\ 0 \end{bmatrix}
\]

Compaction
Discrete Cosine Transform

\[
d_i = \begin{cases} 
    T \frac{1}{\sqrt{2N}} & \text{if } i = 0 \\
    \cos \left( \frac{2(2N) + 1}{2N} \right) & \text{if } i > 0 
\end{cases}
\]

\[
N = 4
\]

\[
D = \begin{bmatrix}
0.65328 & 0.270598 & -270598 & -0.65328 \\
-0.5 & -0.5 & -0.5 & -0.5 \\
-270598 & -0.65328 & 0.65328 & -270598
\end{bmatrix}
\]

Decomposition in Terms of Basis Vectors

\[
\begin{bmatrix}
0.65328 & 0.270598 & -270598 & -0.65328 \\
-0.5 & -0.5 & -0.5 & -0.5 \\
-270598 & -0.65328 & 0.65328 & -270598
\end{bmatrix} \begin{bmatrix}
x_0 \\
x_1 \\
x_2 \\
x_3
\end{bmatrix} = \begin{bmatrix}
x_0 \\
x_1 \\
x_2 \\
x_3
\end{bmatrix}
\]

Block Transform

Each 8x8 block is individually coded

2-Dimensional Block Transform

Block of pixels \( X \)

\[
\begin{bmatrix}
X_{00} & X_{01} & X_{02} & X_{03} \\
X_{10} & X_{11} & X_{12} & X_{13} \\
X_{20} & X_{21} & X_{22} & X_{23} \\
X_{30} & X_{31} & X_{32} & X_{33}
\end{bmatrix}
\]

Transform

\[
A = \begin{bmatrix}
a_{00} & a_{01} & a_{02} & a_{03} \\
a_{10} & a_{11} & a_{12} & a_{13} \\
a_{20} & a_{21} & a_{22} & a_{23} \\
a_{30} & a_{31} & a_{32} & a_{33}
\end{bmatrix}
\]

Transform rows \( f_i = \sum_{k=0}^{3} a_{ik} x_k \)

Transform columns \( c_i = \sum_{k=0}^{3} a_{k} x_{k} = a_{k} x_{0} + \sum_{j=0}^{3} a_{kj} x_{j} = \sum_{j=0}^{3} a_{kj} x_{j} \)

Summary \( C = AXA^T \)
Importance of Coefficients

- The DC coefficient is the most important.
- The AC coefficients become less important as they are farther from the DC coefficient.
- Example Bit Allocation

<table>
<thead>
<tr>
<th>1</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>7</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Compression: 65 bits for 64 pixels = .86 bpp

Quantization

- For a nxn block we construct a nxn matrix Q such that $Q_{ij}$ indicates how many quantization levels to use for coefficient $c_{ij}$.
- Encode $c_{ij}$ with the label

$$s_i = \left[ \frac{c_i}{Q_i} + 0.5 \right]$$

Larger $Q_i$ indicates fewer levels.
- Decode $s_i$ to

$$c_i' = s_i Q_i$$

Example Quantization

- $c = 54.2, Q = 24$
  $$s_i = \left[ \frac{54.2}{24} + 0.5 \right] = 2$$
  $$c_i' = 2 \cdot 24 = 48$$
- $c = 54.2, Q = 12$
  $$s_i = \left[ \frac{54.2}{12} + 0.5 \right] = 5$$
  $$c_i' = 5 \cdot 12 = 60$$
- $c = 54.2, Q = 6$
  $$s_i = \left[ \frac{54.2}{6} + 0.5 \right] = 9$$
  $$c_i' = 9 \cdot 6 = 54$$

Example Quantization Table

<table>
<thead>
<tr>
<th>16</th>
<th>11</th>
<th>10</th>
<th>16</th>
<th>24</th>
<th>40</th>
<th>51</th>
<th>61</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>12</td>
<td>14</td>
<td>19</td>
<td>26</td>
<td>56</td>
<td>60</td>
<td>55</td>
</tr>
<tr>
<td>44</td>
<td>13</td>
<td>16</td>
<td>24</td>
<td>40</td>
<td>57</td>
<td>69</td>
<td>56</td>
</tr>
<tr>
<td>14</td>
<td>17</td>
<td>22</td>
<td>29</td>
<td>51</td>
<td>87</td>
<td>80</td>
<td>62</td>
</tr>
<tr>
<td>33</td>
<td>33</td>
<td>37</td>
<td>56</td>
<td>88</td>
<td>109</td>
<td>103</td>
<td>77</td>
</tr>
<tr>
<td>24</td>
<td>35</td>
<td>55</td>
<td>64</td>
<td>87</td>
<td>104</td>
<td>113</td>
<td>92</td>
</tr>
<tr>
<td>49</td>
<td>64</td>
<td>78</td>
<td>87</td>
<td>103</td>
<td>120</td>
<td>120</td>
<td>101</td>
</tr>
<tr>
<td>72</td>
<td>92</td>
<td>95</td>
<td>98</td>
<td>112</td>
<td>100</td>
<td>103</td>
<td>99</td>
</tr>
</tbody>
</table>

Increase the bit rate = halve the table
Decrease the bit rate = double the table

Zig-Zag Coding

- DC label is coded separately.
- AC labels are usually coded in zig-zag order using a special entropy coding to take advantage of the ordering of the bit allocation (quantization).

JPEG (1987)

- Let $P = [p_{ij}]$, $0 \leq i, j < N$ be an image with $0 \leq p_{ij} < 256$.
- Center the pixels around zero
  $$x_{ij} = p_{ij} - 128$$
- Code 8x8 blocks of P using DCT
- Choose a quantization table.
  - The table depends on the desired quality and is built into JPEG
  - Quantize the coefficients according to the quantization table.
  - The quantization symbols can be positive or negative.
  - Transmit the labels (in a coded way) for each block.
Block Transmission

• DC coefficient
  – DC coefficients don’t change much from block to neighboring block. Hence, their labels change even less.
  – Predictive coding using differences is used to code the DC label.
• AC coefficients
  – Do a zig-zag coding.

Example Block of Labels

Coding Labels

• Categories of labels
  – 1   {0}
  – 2   {-1, 1}
  – 3   {-3,-2,2,3}
  – 4   {-7,-6,-5,-4,4,5 6 7}
• Label is indicated by two numbers C,B
• Examples
  label      C,B
  0         1
  2         3, 2
  -4        4, 3

Coding AC Label Sequence

• A symbol has three parts (Z,C,B)
  – Z for number of zeros preceding a label 0 ≤ Z ≤ 15
  – C for the category of the label
  – B for a C-1 bit number for the actual label
• End of Block symbol (EOB) means the rest of the block is zeros. EOB = (0,0,-)
• Example: (0,3,2)(0,5,7)(0,3,3)(4,2,1)(0,2,1)(2,2,1)(0,0,-)

Notes on Transform Coding

• Video Coding
  – MPEG – uses DCT
  – H.263, H.264 – uses DCT
• Audio Coding
  – MP3 = MPEG 1- Layer 3 uses DCT
• Alternative Transforms
  – Lapped transforms remove some of the blocking artifacts.
  – Wavelet transforms do not need to use blocks at all.