Image Compression Part 2: Variable-rate compression
Variable-rate Compression: Transform-based compression

- As mentioned earlier, we wish to transform image data into a form that reduces statistical correlation.
- We also saw that when images are transformed into the frequency domain using the Fourier Transform:
  - Most of the energy resides in low frequency components.
  - A good approximation of the original image can be reconstructed using a few components.
Variable-rate Compression: Transform-based compression

- Idea: What if we apply an image transform like the Fourier Transform to an image and encode the transform coefficients in a reduced, lossy form instead?
  - Many of the transform coefficients have low associated energies and can be discarded or coarsely quantized with little image distortion
Image Transforms

- Converts images from one representation (e.g., spatial) to another (e.g., frequency)

- **Forward transform**

\[
T(u, v) = \sum_{x=0}^{n-1} \sum_{y=0}^{n-1} g(x, y) r(x, y, u, v)
\]

- **Forward transform kernel**
- **Image**
- **Transform coefficients**
Image Transforms

- Inverse transform

\[ g(x, y) = \sum_{u=0}^{n-1} \sum_{v=0}^{n-1} T(u, v)s(x, y, u, v) \]

- Essentially representing an image using a set of basis functions
Selection of Image Transforms

• Selection of image transform is very important to compression performance as well as computational performance
• Some well-known image transforms
  – Karhunen-Loeve (KL) transform
  – Fourier transform
  – Walsh-Hadamard transform (WHT)
  – Discrete Cosine transform (DCT)
Karhunen-Loeve (KL) transform

- Optimal transform in terms of compression
  - Minimizes mean square error
  - Statistically decorrelated (off-diagonal elements of covariance matrix are zero)

\[
G = \Phi^T F
\]

\[
F = \Phi G \text{ (since } \Phi^{-1} = \Phi^T \text{)}
\]

where \( \Phi^T \Sigma \Phi = \Lambda \)

\[\text{Columns are eigenvectors} \quad \text{Covariance} \quad \text{Eigenvalues}\]
Karhunen-Loeve (KL) transform

- Basis functions essential based on the eigenvectors
- Low-term components (with highest eigenvalues) contain most of the energy
- Advantages:
  - Provides optimal compression performance from an energy compaction perspective
- Disadvantages:
  - Computationally expensive, since the transform is data-dependent and deriving basis functions are non-trivial
Fourier transform

- Transformation kernels

\[ r(x, y, u, v) = e^{-j\frac{2\pi}{n}(ux + vy)} \]

\[ s(x, y, u, v) = \frac{1}{n^2} e^{j\frac{2\pi}{n}(ux + vy)} \]

- Advantages
  - Hardware acceleration available on CPUs

- Disadvantages
  - Relatively poor compression performance
Walsh-Hadamard Transform (WHT)

- Transformation kernels

\[ r(x, y, u, v) = s(x, y, u, v) = \frac{1}{n} \left( \sum_{i=0}^{m-1} [b_i(x) p_i(x) + b_i(y) p_i(v)] \right) \]

where \( n = 2^m \)

- Advantages
  - Computational simple

- Disadvantages
  - Relatively poor compression performance (worse than Fourier transform)
Walsh-Hadamard Transform (WHT)

- Transformation kernels

\[ r(x, y, u, v) = s(x, y, u, v) = \frac{1}{n} \left( -1 \right)^{\sum_{i=0}^{m-1} b_i(x) p_i(x) + b_i(y) p_i(v)} \]

where \( n = 2^m \)

- WH kernels consists of alternating plus (white) and minus (black) 1s in checkerboard pattern
Walsh-Hadamard Transform (WHT)

- **Advantages**
  - Computational simple implementation

- **Disadvantages**
  - Relatively poor compression performance (worse than Fourier transform)

Source: Gonzalez and Woods
Discrete Cosine Transform (DCT)

- Transformation kernels

\[ r(x, y, u, v) = s(x, y, u, v) \]

\[ = \alpha (u) \alpha (v) \cos \left[ \frac{(2x + 1)u\pi}{2n} \right] \cos \left[ \frac{(2y + 1)v\pi}{2n} \right] \]

where

\[ \alpha (u) = \begin{cases} 
\sqrt{\frac{1}{n}} & \text{for } u = 0 \\
\sqrt{\frac{2}{n}} & \text{for } u = 1, 2, \ldots, n - 1 
\end{cases} \]
Discrete Cosine Transform (DCT) Kernel

**FIGURE 8.23**
Discrete-cosine basis functions for $n = 4$. The origin of each block is at its top left.

Source: Gonzalez and Woods
Discrete Cosine Transform (DCT)

- Advantages
  - Computational efficient (easy to implement in hardware)
  - High compression performance (closely approximates performance of KLT for many images)
- Given these benefits, DCT has become an international standard for transform coding
Block Transform Coding

**Issue:** Image transforms such as Fourier Transform are have relatively expensive from both computational and storage perspective
- Must hold all data in memory at once to perform transform
- Must account for all pixels in image when computing transform

**Difficult to implement in consumer-level devices such as DVD players and digital cameras**
Block Transform Coding

**Solution:**
- Break images into a set of smaller sub-images ("blocks")
- Apply image transform on the sub-images independently

**Advantage:**
- The amount of information that needs to be stored to transform a sub-image is small
- Since all operations are independent, they can be performed in parallel to improve computational efficiency
Block Transform Compression Framework

Source: Gonzalez and Woods
How do we deal with color?

- As mentioned before, RGB color space is highly redundant and correlated
  - Reduces compression performance

**Solution**: Use a color space that decorrelates information such as luminance and color

- e.g., Before image transform, convert image from RGB color space to YcbCr
  - Allows luma and chroma channels to be processed independently
Chroma Subsampling

- The human vision system is significantly more sensitive to variations in brightness (luma) than color (chroma).
- **Idea**: reducing the amount of chroma information stored compared to the amount of luma information should have little impact on perceived image quality.
Example: JPEG

- In JPEG, image is converted from RGB to YCbCr
- The resolution of the Cb and Cr channels are reduced
- Commonly, the Cb and Cr channels are sub-sampled by at factor of 2 both horizontally and vertically
Results
Sub-image Construction

- After chroma subsampling, the individual channels are divided into a set of $n \times n$ sub-images.
- Generally, compression performance increases as sub-image size increases.
- However, computational complexity increases as sub-image size increases.
- Drawing a balance between compression performance and compression efficiency, 8x8 and 16x16 sub-image sizes are used.
Reconstruction Error vs. Sub-image Size

- 75% of coefficients are truncated

Source: Gonzalez and Woods
Reconstruction Quality vs. Sub-image Size

**FIGURE 8.27** Approximations of Fig. 8.27(a) using 25% of the DCT coefficients and (b) $2 \times 2$ subimages, (c) $4 \times 4$ subimages, and (d) $8 \times 8$ subimages. The original image in (a) is a zoomed section of Fig. 8.9(a).

Source: Gonzalez and Woods
Quantization

- As mentioned earlier, the human vision system is much more sensitive to variations in low frequency components than high frequency components.
- Also, much of the energy is packed in the low frequency components.
- **Idea**: high frequency components can be represented coarsely ("quantized") without perceptually noticeable degradation in image quality.
Quantization

• **Steps**
  - Transform image $f$ from the spatial representation to $T$ in the transform domain
  - Quantize $T$ based on a quantization matrix $Z$ designed for the human vision system based on perceptual importance

$$\hat{T}(u, v) = \text{round} \left[ \frac{T(u, v)}{Z(u, v)} \right]$$
Quantization Matrix (JPEG)

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>11</td>
<td>10</td>
<td>16</td>
<td>24</td>
<td>40</td>
<td>51</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>14</td>
<td>19</td>
<td>26</td>
<td>58</td>
<td>60</td>
</tr>
<tr>
<td>14</td>
<td>13</td>
<td>16</td>
<td>24</td>
<td>40</td>
<td>57</td>
<td>69</td>
</tr>
<tr>
<td>14</td>
<td>17</td>
<td>22</td>
<td>29</td>
<td>51</td>
<td>87</td>
<td>80</td>
</tr>
<tr>
<td>18</td>
<td>22</td>
<td>37</td>
<td>56</td>
<td>68</td>
<td>109</td>
<td>103</td>
</tr>
<tr>
<td>24</td>
<td>35</td>
<td>55</td>
<td>64</td>
<td>81</td>
<td>104</td>
<td>113</td>
</tr>
<tr>
<td>49</td>
<td>64</td>
<td>78</td>
<td>87</td>
<td>103</td>
<td>121</td>
<td>120</td>
</tr>
<tr>
<td>72</td>
<td>92</td>
<td>95</td>
<td>98</td>
<td>112</td>
<td>100</td>
<td>103</td>
</tr>
</tbody>
</table>

Source: Gonzalez and Woods
Effect of Quantization Level on Image Quality

**FIGURE 8.31** Approximations of Fig. 8.9(a) using the DCT and normalization array of Fig. 8.30(b): (a) Z, (b) 2Z, (c) 4Z, (d) 8Z, (e) 16Z, and (f) 32Z.

Source: Gonzalez and Woods
Observations

• As quantization increases
  – fine image detail starts to be lost (e.g., mouth and feathers start to degrade until completely disappearing)
  – Blocking artifacts (i.e., visible boundaries between sub-images) becomes increasingly prominent
• However, uniform regions with little detail are significantly less affected by quantization
Adaptive Quantization

- High quantization are perceptually acceptable in uniform regions
- Low quantization is needed in regions with structural detail
- **Idea:** Adjust degree of quantization based on amount of image detail within a sub-image
  - Measure level of image detail (e.g., variance) of the sub-image
  - Decrease quantization for sub-images with high image detail
Example

Fixed quantization  Adaptive quantization
Predictive Coding

- Images and videos contain a large amount of spatial and temporal redundancy
- Pixels in an image or video frame should be reasonably predicted by other pixels in
  - The same image (**intra-frame prediction**)
  - Adjacent frames (**inter-frame prediction**)
Intra-frame Predictive Coding

- For a sub-image $f$, find the sub-image $p$ that is most similar to $f$ (block matching)
- One approach is to find the sub-image that minimizes the mean absolute distortion (MAD)

$$MAD(x, y) = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} |f(x+i, y+j) - f(x+i+dx, x+j+dy)|$$

- Usually performed on the luminance channel
- Encode and store vector $(dx, dy)$
Intra-frame Predictive Coding

- Calculate the error residual between the two sub-images

\[ e(x, y) = f(x + i, y + j) - f(x + i + dx, x + j + dy) \]

where \( i, j \) spans the dimension of the sub-image

- Transform prediction error residual with image transform and quantized
Inter-frame Prediction Coding (Motion Compensation)

- Similar to intra-frame coding, but instead of within the same image, the prediction coding is performed between frames

Source: Gonzalez and Woods
Results using Inter-frame Prediction Coding

**FIGURE 8.37** (a) and (b) Two views of Earth that are thirteen frames apart in an orbiting space shuttle video. (c) A prediction error image without motion compensation. (d) The prediction residual with motion compensation. (e) The motion vectors associated with (d). The white dots in (d) represent the arrow heads of the motion vectors that are depicted. (Original images courtesy of NASA.)

Source: Gonzalez and Woods