Chapter 9

Image Compression Standards

9.1 The JPEG Standard
9.2 The JPEG2000 Standard
9.3 The JPEG LS Standard
9.4 Bi level Image Compression Standards
9.5 Further Exploration
9.1 The JPEG Standard

- JPEG is an image compression standard that was developed by the “Joint Photographic Experts Group”. JPEG was formally accepted as an international standard in 1992.

- JPEG is a lossy image compression method. It employs a transform coding method using the DCT (Discrete Cosine Transform).

- An image is a function of $i$ and $j$ (or conventionally $x$ and $y$) in the spatial domain.

The 2D DCT is used as one step in JPEG in order to yield a frequency response which is a function $F(u,v)$ in the spatial frequency domain, indexed by two integers $u$ and $v$. 
Observations for JPEG Image Compression

• The effectiveness of the DCT transform coding method in JPEG relies on 3 major observations:

Observation 1: Useful image contents change relatively slowly across the image, i.e., it is unusual for intensity values to vary widely several times in a small area, for example, within an $8 \times 8$ image block.

• much of the information in an image is repeated, hence “spatial redundancy”.
Observations for JPEG Image Compression (cont’d)

Observation 2: Psychophysical experiments suggest that humans are much less likely to notice the loss of very high spatial frequency components than the loss of lower frequency components.

- the spatial redundancy can be reduced by largely reducing the high spatial frequency contents.

Observation 3: Visual acuity (accuracy in distinguishing closely spaced lines) is much greater for gray (“black and white”) than for color.

- chroma subsampling (4:2:0) is used in JPEG.
Fig. 9.1: Block diagram for JPEG encoder.
9.1.1 Main Steps in JPEG Image Compression

- Transform RGB to YIQ or YUV and subsample color.
- DCT on image blocks.
- Quantization.
- Zig-zag ordering and run-length encoding.
- Entropy coding.
**DCT on image blocks**

- Each image is divided into $8 \times 8$ blocks. The 2D DCT is applied to each block image $f(i,j)$, with output being the DCT coefficients $F(u,v)$ for each block.

- Using blocks, however, has the effect of isolating each block from its neighboring context. This is why JPEG images look choppy ("blocky") when a high compression ratio is specified by the user.
Quantization

\[ \hat{F}(u,v) = \text{round} \left( \frac{F(u,v)}{Q(u,v)} \right) \]  

(9.1)

- \( F(u,v) \) represents a DCT coefficient, \( Q(u,v) \) is a “quantization matrix” entry, and \( \hat{F}(u,v) \) represents the quantized DCT coefficients which JPEG will use in the succeeding entropy coding.

- The quantization step is the main source for loss in JPEG compression.

- The entries of \( Q(u,v) \) tend to have larger values towards the lower right corner. This aims to introduce more loss at the higher spatial frequencies — a practice supported by Observations 1 and 2.

- Table 9.1 and 9.2 show the default \( Q(u,v) \) values obtained from psychophysical studies with the goal of maximizing the compression ratio while minimizing perceptual losses in JPEG images.
### Table 9.1  The Luminance 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>58</td>
<td>60</td>
<td>55</td>
</tr>
<tr>
<td>14</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>18</td>
<td>22</td>
<td>37</td>
<td>56</td>
<td>68</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>81</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>121</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>

### Table 9.2  The Chrominance Quantization Table

<table>
<thead>
<tr>
<th>17</th>
<th>18</th>
<th>24</th>
<th>47</th>
<th>99</th>
<th>99</th>
<th>99</th>
<th>99</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>21</td>
<td>26</td>
<td>66</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
</tr>
<tr>
<td>24</td>
<td>26</td>
<td>56</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
</tr>
<tr>
<td>47</td>
<td>66</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
<td>99</td>
</tr>
</tbody>
</table>
An 8 × 8 block from the Y image of ‘Lena’

\[
\begin{array}{cccccccccccc}
200 & 202 & 189 & 188 & 189 & 175 & 175 & 175 & 515 & 65 & -12 & 4 & 1 & 2 & -8 & 5 \\
200 & 203 & 198 & 188 & 189 & 182 & 178 & 175 & -16 & 3 & 2 & 0 & 0 & -11 & -2 & 3 \\
203 & 200 & 200 & 195 & 200 & 187 & 185 & 175 & -12 & 6 & 11 & -1 & 3 & 0 & 1 & -2 \\
200 & 200 & 200 & 200 & 200 & 190 & 187 & 175 & 0 & -3 & -1 & 0 & 4 & 1 & -1 & 0 \\
205 & 200 & 199 & 200 & 191 & 187 & 187 & 175 & 3 & -2 & -3 & 3 & 3 & -1 & -1 & 3 \\
\end{array}
\]

\[f(i, j)\] \hspace{4in} \[F(u, v)\]

Fig. 9.2: JPEG compression for a smooth image block.
Fig. 9.2 (cont’d): JPEG compression for a smooth image block.
Another $8 \times 8$ block from the $Y$ image of ‘Lena’

<table>
<thead>
<tr>
<th>$f(i,j)$</th>
<th>$F(u,v)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>70 70 100 70 87 87 150 187</td>
<td>-80 -40 89 -73 44 32 53 -3</td>
</tr>
<tr>
<td>85 100 96 79 87 154 87 113</td>
<td>-135 -59 -26 6 14 -3 -13 -28</td>
</tr>
<tr>
<td>100 85 116 79 70 87 86 196</td>
<td>47 -76 66 -3 -108 -78 33 59</td>
</tr>
<tr>
<td>136 69 87 200 79 71 117 96</td>
<td>-2 10 -18 0 33 11 -21 1</td>
</tr>
<tr>
<td>161 70 87 200 103 71 96 113</td>
<td>-1 -9 -22 8 32 65 -36 -1</td>
</tr>
<tr>
<td>161 123 147 133 113 113 85 161</td>
<td>5 -20 28 -46 3 24 -30 24</td>
</tr>
<tr>
<td>146 147 175 100 103 103 163 187</td>
<td>6 -20 37 -28 12 -35 33 17</td>
</tr>
<tr>
<td>156 146 189 70 113 161 163 197</td>
<td>-5 -23 33 -30 17 -5 -4 20</td>
</tr>
</tbody>
</table>

Fig. 9.3: JPEG compression for a textured image block.
\[
\begin{array}{cccccc}
-5 & -4 & 9 & -5 & 2 & 1 & 1 & 0 \\
-11 & -5 & -2 & 0 & 1 & 0 & 0 & -1 \\
3 & -6 & 4 & 0 & -3 & -1 & 0 & 1 \\
0 & 1 & -1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\
0 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{array}
\]

\[\hat{F}(u, v)\]

\[
\begin{array}{cccc}
70 & 60 & 106 & 94 \\
85 & 101 & 85 & 75 \\
98 & 99 & 92 & 102 \\
132 & 53 & 111 & 180 \\
173 & 57 & 114 & 207 \\
164 & 123 & 131 & 133 \\
141 & 159 & 169 & 73 \\
150 & 141 & 195 & 79 \\
\end{array}
\]

\[\tilde{f}(i, j)\]

\[
\begin{array}{cccc}
-80 & -44 & 90 & -80 \\
-132 & -60 & -28 & 0 \\
42 & -78 & 64 & -120 \\
0 & 17 & -22 & 0 \\
0 & 0 & -37 & 0 \\
0 & -35 & 55 & -64 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{array}
\]

\[\hat{F}(u, v)\]

\[
\begin{array}{cccc}
0 & 10 & -6 & -24 \\
0 & -1 & 11 & 4 \\
2 & -14 & 24 & -23 \\
4 & 16 & -24 & 20 \\
-12 & 13 & -27 & -7 \\
-3 & 0 & 16 & -2 \\
5 & -12 & 6 & -27 \\
6 & 5 & -6 & -9 \\
\end{array}
\]

\[
\epsilon(i, j) = f(i, j) - \tilde{f}(i, j)
\]

Fig. 9.3 (cont’d): JPEG compression for a textured image block.
Run-length Coding (RLC) on AC coefficients

- RLC aims to turn the $\hat{F}(u, v)$ values into sets $\{\text{#-zeros-to-skip, next non-zero value}\}$.

- To make it most likely to hit a long run of zeros: a zig-zag scan is used to turn the $8 \times 8$ matrix $\hat{F}(u, v)$ into a 64-vector.

Fig. 9.4: Zig-Zag Scan in JPEG.
DPCM on DC coefficients

- The DC coefficients are coded separately from the AC ones. *Differential Pulse Code Modulation (DPCM)* is the coding method.

- If the DC coefficients for the first 5 image blocks are 150, 155, 149, 152, 144, then the DPCM would produce 150, 5, -6, 3, -8, assuming $d_i = DC_{i+1} - DC_i$, and $d_0 = DC_0$. 
Entropic Coding

- The DC and AC coefficients finally undergo an entropy coding step to gain a possible further compression.

- Use DC as an example: each DPCM coded DC coefficient is represented by (SIZE, AMPLITUDE), where SIZE indicates how many bits are needed for representing the coefficient, and AMPLITUDE contains the actual bits.

- In the example we’re using, codes 150, 5, –6, 3, –8 will be turned into
  
  \[(8, 10010110), (3, 101), (3, 001), (2, 11), (4, 0111)\].

- SIZE is Huffman coded since smaller SIZEs occur much more often. AMPLITUDE is not Huffman coded, its value can change widely so Huffman coding has no appreciable benefit.
Table 9.3  Baseline entropy coding details – size category.

<table>
<thead>
<tr>
<th>SIZE</th>
<th>AMPLITUDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-1, 1</td>
</tr>
<tr>
<td>2</td>
<td>-3, -2, 2, 3</td>
</tr>
<tr>
<td>3</td>
<td>-7..-4, 4..7</td>
</tr>
<tr>
<td>4</td>
<td>-15..-8, 8..15</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>-1023..-512, 512..1023</td>
</tr>
</tbody>
</table>
9.1.2 Four Commonly Used JPEG Modes

- Sequential Mode — the default JPEG mode, implicitly assumed in the discussions so far. Each graylevel image or color image component is encoded in a single left-to-right, top-to-bottom scan.

- Progressive Mode.

- Hierarchical Mode.

- Lossless Mode — discussed in Chapter 7, to be replaced by JPEG-LS (Section 9.3).
Progressive Mode

Progressive JPEG delivers low quality versions of the image quickly, followed by higher quality passes.

1. **Spectral selection**: Takes advantage of the “spectral” (spatial frequency spectrum) characteristics of the DCT coefficients: higher AC components provide detail information.

   Scan 1: Encode DC and first few AC components, e.g., AC1, AC2.

   Scan 2: Encode a few more AC components, e.g., AC3, AC4, AC5.

   \vdots

   Scan k: Encode the last few ACs, e.g., AC61, AC62, AC63.
Progressive Mode (Cont’d)

2. **Successive approximation**: Instead of gradually encoding spectral bands, all DCT coefficients are encoded simultaneously but with their most significant bits (MSBs) first.

   Scan 1: Encode the first few MSBs, e.g., Bits 7, 6, 5, 4.

   Scan 2: Encode a few more less significant bits, e.g., Bit 3.

   : 

   Scan m: Encode the least significant bit (LSB), Bit 0.
Hierarchical Mode

• The encoded image at the lowest resolution is basically a compressed low-pass filtered image, whereas the images at successively higher resolutions provide additional details (differences from the lower resolution images).

• Similar to Progressive JPEG, the Hierarchical JPEG images can be transmitted in multiple passes progressively improving quality.
Fig. 9.5: Block diagram for Hierarchical JPEG.
Encoder for a Three-level Hierarchical JPEG

1. Reduction of image resolution:
   Reduce resolution of the input image $f$ (e.g., $512 \times 512$) by a factor of 2 in each dimension to obtain $f_2$ (e.g., $256 \times 256$). Repeat this to obtain $f_4$ (e.g., $128 \times 128$).

2. Compress low-resolution image $f_4$:
   Encode $f_4$ using any other JPEG method (e.g., Sequential, Progressive) to obtain $F_4$.

3. Compress difference image $d_2$:
   (a) Decode $F_4$ to obtain $\tilde{f}_4$. Use any interpolation method to expand $\tilde{f}_4$ to be of the same resolution as $f_2$ and call it $E(\tilde{f}_4)$.
   (b) Encode difference $d_2 = f_2 - E(\tilde{f}_4)$ using any other JPEG method (e.g., Sequential, Progressive) to generate $D_2$.

4. Compress difference image $d_1$:
   (a) Decode $D_2$ to obtain $\tilde{d}_2$; add it to $E(\tilde{f}_4)$ to get $\tilde{f}_2 = E(\tilde{f}_4) + \tilde{d}_2$ which is a version of $f_2$ after compression and decompression.
   (b) Encode difference $d_1 = f - E(\tilde{f}_2)$ using any other JPEG method (e.g., Sequential, Progressive) to generate $D_1$. 
Decoder for a Three-level Hierarchical JPEG

1. Decompress the encoded low-resolution image $F_4$:
   
   - Decode $F_4$ using the same JPEG method as in the encoder to obtain $\tilde{f}_4$.

2. Restore image $\tilde{f}_2$ at the intermediate resolution:
   
   - Use $E(\tilde{f}_4) + \tilde{d}_2$ to obtain $\tilde{f}_2$.

3. Restore image $\tilde{f}$ at the original resolution:
   
   - Use $E(\tilde{f}_2) + \tilde{d}_1$ to obtain $\tilde{f}$.
9.1.3 A Glance at the JPEG Bitstream

Fig. 9.6: JPEG bitstream.
9.2 The JPEG2000 Standard

• Design Goals:
  – To provide a better rate-distortion tradeoff and improved subjective image quality.
  – To provide additional functionalities lacking in the current JPEG standard.

• The JPEG2000 standard addresses the following problems:
  – Lossless and Lossy Compression: There is currently no standard that can provide superior lossless compression and lossy compression in a single bitstream.
- **Low Bit-rate Compression**: The current JPEG standard offers excellent rate-distortion performance in mid and high bit-rates. However, at bit-rates below 0.25 bpp, subjective distortion becomes unacceptable. This is important if we hope to receive images on our web-enabled ubiquitous devices, such as web-aware wristwatches and so on.

- **Large Images**: The new standard will allow image resolutions greater than 64K by 64K without tiling. It can handle image size up to $2^{32} - 1$.

- **Single Decompression Architecture**: The current JPEG standard has 44 modes, many of which are application specific and not used by the majority of JPEG decoders.
- **Transmission in Noisy Environments**: The new standard will provide improved error resilience for transmission in noisy environments such as wireless networks and the Internet.

- **Progressive Transmission**: The new standard provides seamless quality and resolution scalability from low to high bit-rate. The target bit-rate and reconstruction resolution need not be known at the time of compression.

- **Region of Interest Coding**: The new standard allows the specification of Regions of Interest (ROI) which can be coded with superior quality than the rest of the image. One might like to code the face of a speaker with more quality than the surrounding furniture.
- **Computer Generated Imagery**: The current JPEG standard is optimized for natural imagery and does not perform well on computer generated imagery.

- **Compound Documents**: The new standard offers metadata mechanisms for incorporating additional non-image data as part of the file. This might be useful for including text along with imagery, as one important example.

  - In addition, JPEG2000 is able to handle up to 256 channels of information whereas the current JPEG standard is only able to handle three color channels.
9.5 Further Explorations

• **Text books:**
  
  – *The JPEG Still Image Compression Standard* by Pennebaker and Mitchell
  
  – *JPEG2000: Image Compression Fundamentals, Standards, and Practice* by Taubman and Marcellin
  

• Interactive JPEG demo, and comparison of JPEG and JPEG2000

• **Web sites:** [Link to Further Exploration for Chapter 9.](#) including:
  
  – JPEG and JPEG2000 links, source code, etc.
  
  – Original paper for the LOCO-I algorithm
  
  – Introduction and source code for JPEG-LS, JBIG, JBIG2