Compression

Key Concepts

Lossless vs. lossy compression
Kolmogorov complexity
Entropy and Huffman coding
JPEG / Discrete cosine transform (DCT)
JPEG2000 / Haar Wavelets
Image and Video Data Rates

One full screen Image
- $1024 \times 768 \times 24b = \sim 2.5MB$

DVD
- $720 \times 480 \times 24b \times 30f/s = \sim 30 \text{ MB/s}$

High Definition DVD
- $1920 \times 1080 \times 24b \times 30f/s = \sim 178\text{MB/s}$

Film
- $4000 \times 3000 \times 36b \times 30f/s = \sim 1.5\text{GB/s}$
- 8 TB for a 90 minute movie!

Lossless vs. Lossy Compression

Lossless
- All information stored
- Original can be reconstructed exactly

Lossy
- Some information discarded
- Much higher compression ratios possible

Strategies
- Discard information that rarely occurs
- Discard information humans won’t notice
Kolmogorov Complexity

What is the shortest program that can generate this fractal image?

64 Byte C program

\[
\begin{align*}
  z &= z_0 = \text{Complex}(x, y) \\
  \text{for}(i=0; i<1000; i++) \\
  z &= z^2 + c \\
  \text{setpixel}(x, y, |z-z_0|)
\end{align*}
\]

24 Bytes of parameters

Re(c), Im(c),
xmin, xmax,
ymin, ymax

17 KB JPEG

Lossless Compression
Image Entropy

An image has low entropy if its pixel values are predictable.
An image has high entropy if its pixel values are random (unpredictable).

Example

Alphabet: A, B, C, D
Frequencies: ¼, ¼, ¼, ¼
Code: 00, 01, 10, 11

ABCDBDCA = 8 characters
00 01 10 11 01 11 00 11 = 16 bits

¼ * 2 + ¼ * 2 + ¼ * 2 + ¼ * 2 = 2 bits/ch
Example

Alphabet: A, B, C, D
Frequencies: $\frac{1}{2}$, $\frac{1}{4}$, 1/8, 1/8
Code: 0, 10, 110, 111

ABACADAB = 8 characters
0 10 0 110 0 111 0 10 = 14 bits

$\frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{8} \times 3 + \frac{1}{8} \times 3 = 1 \frac{3}{4}$ bits/ch

Definition of Entropy

Given a random variable $X$ with possible values $x_1, x_2, ... x_n$, each with probability $p(x_1), p(x_2), ... p(x_n)$, the entropy is given as:

$$H(X) = \sum_{i=1}^{n} p(x_i) \log_2 \left( \frac{1}{p(x_i)} \right)$$

$$= - \sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

$\log 1/p(x_i)$ is the number of bits to code $x_i$

$H$ is the average number of bits
Huffman Coding

Input
- Symbols and their probabilities

Output
- Binary code for each symbol

Less frequent values have longer codes than more frequent values

A (.10)
B (.15)
C (.30)
D (.16)
E (.29)
Huffman Coding

AB (.25)

A (.10) B (.15)

0 1

C (.30)
D (.16)
E (.29)

AB (.25)

A (.10) B (.15)

0 1

ABD (.41)

A (.10) B (.15) D (.16)

0 1

C (.30)
E (.29)
Huffman Coding

ABD (.41)

0 1

AB (.25) D (.16) C (.30) E (.29) CE (.59)

0 1

A (.10) B (.15) ABD CE (ABD CE)

0 1

ABCDE (1.0)

ABD (.41)

0 1

AB (.25) D (.16) C (.30) E (.29)

0 1

A (.10) B (.15)
Huffman Coding

Algorithm

- Create a node for each value
- Repeat until there is only one node left:
  - Merge the two nodes with lowest probabilities into a new node whose probability is the sum of its children
- The path from the root node to the leaf node is the code for each value
Transform Coding

If you transform the input data, it may be much more predictable.

For example, taking differences:

\[ X = 2 \ 2 \ 2 \ 2 \ 3 \ 4 \ 5 \ 6 \ 6 \ 6 \ 6 \]

Difference Operator

\[ D(X) = 2 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0 \]

\[ H(X) = 2.0049 \]

\[ H(D(X)) = 1.3222 \]
Two Observations on Images

In natural images, low frequencies are more common than high frequencies
Therefore, more bits should be allocated to low frequencies

The human visual system is less sensitive to high frequencies
Therefore, distortion or errors in high frequencies are harder to see (to an extent)
Discrete Cosine Basis

\[
\cos \left[ \pi \frac{k}{N} \left( x + \frac{1}{2} \right) \right]
\]
2D Discrete Cosine Basis (8 by 8)

\[ \cos \left( \frac{\pi i}{N} \left( x + \frac{1}{2} \right) \right) \times \cos \left( \frac{\pi j}{N} \left( y + \frac{1}{2} \right) \right) \]

Discrete Cosine Transform (DCT)

Image space

Frequency space
Lossy Compression

262,144 terms

Largest terms

43384/262144 = 16%
Lossy Compression

\[ \frac{8353}{262144} = 3.2\% \]

Largest terms

Quantization

**Def:** Quantization is the process of mapping a high precision value to a low precision value

- **256 levels**
  - 8 bits/pixel

- **4 levels**
  - 2 bits/pixel
More Bits to Lower Frequencies

Lower precision buy dividing

DCT

\[
\begin{bmatrix}
-415 & -30 & -61 & 27 & 56 & -20 & -2 & 0 \\
4 & -22 & -61 & 10 & 13 & -7 & -9 & 5 \\
-47 & 7 & 77 & -25 & -29 & 10 & 5 & -6 \\
-49 & 12 & 34 & -15 & -10 & 6 & 2 & 2 \\
12 & -7 & -13 & -4 & -2 & 2 & -3 & 3 \\
-8 & 3 & 2 & -6 & -2 & 1 & 4 & 2 \\
-1 & 0 & 0 & -2 & -1 & -3 & 4 & -1 \\
0 & 0 & -1 & -4 & -1 & 0 & 1 & 2
\end{bmatrix}
\]

Quantization Matrix

\[
\begin{bmatrix}
16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\
12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\
14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\
14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\
18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\
24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\
49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\
72 & 92 & 95 & 98 & 112 & 100 & 103 & 99
\end{bmatrix}
\]

\[
\frac{1}{/}
\]

Quantized DCT

\[
\begin{bmatrix}
-26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\
0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\
-3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\
-4 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

Quantization Introduces Error

\[
error = \text{abs(original – compressed)} \times 8
\]
**Measuring Error**

Original image $I$, quantized (compressed) image $I'$

**Mean-Squared-Error (MSE)**

\[
MSE = \frac{1}{N} \sum_{i=0}^{N} (I(i) - I'(i))^2
\]

**Peak Signal (max) to Noise Ratio (error) (PSNR)**

\[
PSNR = 10 \log_{10} \left( \frac{I_{max}^2}{MSE} \right)
\]

*Measure error vs. compression ratio*

---

**JPEG: Block Artifacts**

Note “blockiness”

Original | Compressed
Cosine vs. Haar Basis

Cosines

Haar Wavelets
Haar Wavelets

Scaling function = average

Wavelet = difference

Transform:

\[ S = \frac{(A + B)}{2} \]
\[ D = \frac{(A - B)}{2} \]

Reverse transform:

\[ A = S + D \]
\[ B = S - D \]
Haar Wavelets

\[
\begin{array}{cccccccc}
6 & 8 & 5 & 9 & 5 & 5 & 6 & 6 \\
7 & 7 & 5 & 6 & -1 & -2 & 0 & 0 \\
\end{array}
\]

Difference

1 Transform Step

Averages
Smoothed version of signal
Lower resolution image

Differences
Local, high frequencies
Details missing from low resolution part
Haar Wavelets

\[
\begin{array}{cccccccc}
6 & 8 & 5 & 9 & 5 & 5 & 6 & 6 \\
\end{array}
\]

\[
\begin{array}{cccccccc}
7 & 7 & 5 & 6 & -1 & -2 & 0 & 0 \\
\end{array}
\]

\[
\begin{array}{cccccccc}
7 & 5.5 & -1 & -2 & 0 & 0 \\
\end{array}
\]
Haar Wavelets

\[
\begin{array}{ccccccccc}
6 & 8 & 5 & 9 & 5 & 5 & 6 & 6 \\
\end{array}
\]

\[
\begin{array}{ccccccccc}
7 & 7 & 5 & 6 & -1 & -2 & 0 & 0 \\
\end{array}
\]

\[
\begin{array}{ccccccccc}
7 & 5.5 & 0 & -0.5 & -1 & -2 & 0 & 0 \\
\end{array}
\]
Haar Wavelets

6 8 5 9 5 5 6 6

7 7 5 6 -1 -2 0 0

7 5.5 0 -0.5 -1 -2 0 0

6.25 0.75 0 -0.5 -1 -2 0 0

Haar Wavelets

6 8 5 9 5 5 6 6

7 7 5 6 -1 -2 0 0

7 5.5 0 -0.5 -1 -2 0 0

6.25 0.75 0 -0.5 -1 -2 0 0
Haar Wavelets

\[ \begin{array}{cccccccc}
6 & 8 & 5 & 9 & 5 & 5 & 6 & 6 \\
\end{array} \]

Full Transform

\[ \begin{array}{ccccccc}
6.25 & .75 & 0 & -.5 & -1 & -2 & 0 & 0 \\
\end{array} \]

- High Resolution Details
- Medium Res. Details
- Low Resolution Details
- Average Value

2D: Standard Decomposition

Stollnitz, Derose, Salesin, Wavelets for computer graphics
Wavelet Compression

1. Transform to wavelet basis
2. Quantize results to compress image
3. Huffman compress the quantized values

Works better than DCT, fewer artifacts
What else needs compression?

Video (MPEG-2 and MPEG-4)
Textures

Geometry (including animated geometry)
...

Things to Remember

Lossless
- Compression depends on the entropy

Lossy
- Visual system less sensitive to high frequencies
- Distribute error to higher frequencies
- Measure error vs. number of bits (PSNR)

Different basis functions
- JPEG / Discrete cosine transform (DCT) - good
- JPEG2000 / Wavelets - better