CSEP 590
Data Compression
Autumn 2007

Wavelet Transform Coding
PACW
Wavelet Transform

- Wavelet Transform
  - A family of transformations that filters the data into low resolution data plus detail data.
Wavelet Transformed Barbara (Enhanced)

Low resolution subband

Detail subbands
Wavelet Transformed Barbara (Actual)

most of the details are small so they are very dark.
Wavelet Transform Compression

Wavelet coder transmits wavelet transformed image in bit plane order with the most significant bits first.
Bit Planes of Coefficients

Coefficients are normalized between $-1$ and $1$
Why Wavelet Compression Works

• Wavelet coefficients are transmitted in bit-plane order.
  – In most significant bit planes most coefficients are 0 so they can be coded efficiently.
  – Only some of the bit planes are transmitted. This is where fidelity is lost when compression is gained.

• Natural progressive transmission

  compressed bit planes
  ...

  truncated compressed bit planes
The more bit planes of the wavelet transformed image that are sent the higher the fidelity.
Wavelet Coding Methods

- **EZW** - Shapiro, 1993
  - Embedded Zerotree coding.
- **SPIHT** - Said and Pearlman, 1996
  - Set Partitioning in Hierarchical Trees coding. Also uses “zerotrees”.
- **ECECOW** - Wu, 1997
  - Uses arithmetic coding with context.
- **EBCOT** – Taubman, 2000
  - Uses arithmetic coding with different context.
- **JPEG 2000** – new standard based largely on EBCOT
- **GTW** – Hong, Ladner 2000
  - Uses group testing which is closely related to Golomb codes
- **PACW** - Ladner, Askew, Barney 2003
  - Like GTW but uses arithmetic coding
A wavelet transform decomposes the image into a low resolution version and details. The details are typically very small so they can be coded in very few bits.
One-Dimensional Average Transform (1)

How do we represent two data points at lower resolution?

\[
\frac{x+y}{2} \quad \frac{y-x}{2}
\]

average detail
One-Dimensional Average Transform

\[
\frac{x+y}{2} = L \\
\frac{y-x}{2} = H
\]

*low pass filter*

\[
x = L - H \\
y = L + H
\]

*high pass filter*

Transform

detail

Inverse Transform
One-Dimensional Average Transform (3)

Note that the low resolution version and the detail together have the same number of values as the original.
One-Dimensional Average Transform

(4)

\[ B[i] = \frac{1}{2} A[2i] + \frac{1}{2} A[2i+1], \quad 0 \leq i < \frac{n}{2} \]

\[ B[n/2 + i] = -\frac{1}{2} A[2i] + \frac{1}{2} A[2i+1], \quad 0 \leq i < \frac{n}{2} \]

\[ L = B[0..n/2-1] \]
\[ H = B[n/2..n-1] \]
One-Dimensional Average Inverse Transform

\[ A[2i] = B[i] - B[n/2 + i], \quad 0 \leq i < \frac{n}{2} \]

\[ A[2i + 1] = B[i] + B[n/2 + i], \quad 0 \leq i < \frac{n}{2} \]
Two Dimensional Transform (1)

Transform each row

Transform each column in L and H

3 detail subbands

low resolution subband

L
H
LL LH HL HH

horizontal transform
vertical transform
Two Dimensional Transform (1)

Transform each row in LL

2 levels of transform gives 7 subbands.

k levels of transform gives $3k + 1$ subbands.
Two Dimensional Average Transform

horizontal transform

vertical transform

negative value
Wavelet Transformed Image

2 levels of wavelet transform

1 low resolution subband

6 detail subbands
Wavelet Transform Details

• Conversion to reals.
  – Convert gray scale to floating point.
  – Convert color to Y U V and then convert each band to floating point. Compress separately.

• After several levels (3-8) of transform we have a matrix of floating point numbers called the wavelet transformed image (coefficients).
Wavelet Transforms

• Technically wavelet transforms are special kinds of linear transformations. Easiest to think of them as filters.
  – The filters depend only on a constant number of values. (bounded support)
  – Preserve energy (norm of the pixels = norm of the coefficients)
  – Inverse filters also have bounded support.

• Well-known wavelet transforms
  – Haar – like the average but orthogonal to preserve energy. Not used in practice.
  – Daubechies 9/7 – biorthogonal (inverse is not the transpose). Most commonly used in practice.
Haar Filters

low pass $= \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}$

high pass $= -\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}$

low pass $B[i] = \frac{1}{\sqrt{2}} A[2i] + \frac{1}{\sqrt{2}} A[2i+1], \quad 0 \leq i < \frac{n}{2}$

high pass $B[n/2 + i] = -\frac{1}{\sqrt{2}} A[2i] + \frac{1}{\sqrt{2}} A[2i+1], \quad 0 \leq i < \frac{n}{2}$

Want the sum of squares of the filter coefficients $= 1$
Daubechies 9/7 Filters

low pass filter

high pass filter

low pass \[ B[i] = \sum_{j=-4}^{4} h_j A[2i + j], \quad 0 \leq i < \frac{n}{2} \]

high pass \[ B[n/2 + i] = \sum_{j=-3}^{3} g_j A[2i + j], \quad 0 \leq i < \frac{n}{2} \]

reflection used near boundaries
Linear Time Complexity of 2D Wavelet Transform

• Let $n =$ number of pixels and let $b$ be the number of coefficients in the filters.

• One level of transform takes time
  – $O(bn)$

• $k$ levels of transform takes time proportional to
  – $bn + \frac{bn}{4} + \ldots + \frac{bn}{4^{k-1}} < \frac{4}{3}bn$

• The wavelet transform is linear time when the filters have constant size.
  – The point of wavelets is to use constant size filters unlike many other transforms.
Wavelet coder transmits wavelet transformed image in bit plane order with the most significant bits first.
Bit-Plane Coding

• Normalize the coefficients to be between $-1$ and 1
• Transmit one bit-plane at a time
• For each bit-plane
  – Significance pass: Find the newly significant coefficients, transmit their signs.
  – Refinement pass: transmit the bits of the known significant coefficients.
Divide into Bit-Planes

Coefficients

+0010  -0011
+0001  -1000
+0101  +0111

Sign Plane

1
2
·
·
Significant Coefficients
Significant Coefficients

![Graph showing significant coefficients]

- Magnitude
- Bit-plane 2 threshold
- Coefficients
Significance & Refinement Passes

- Code a bit-plane in two passes
  - Significance pass
    - codes previously insignificant coefficients
    - also codes sign bit
  - Refinement pass
    - refines values for previously significant coefficients
- Main idea:
  - Significance-pass bits likely to be 0;
  - Refinement-pass bit are not

<table>
<thead>
<tr>
<th>#</th>
<th>value</th>
<th>refinement bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>010010010110</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>001011011110</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>000001001001</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>000000010110</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>000100111101</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>000000100101</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>101101110101</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>010010011111</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>001011101101</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>000010100101</td>
<td></td>
</tr>
</tbody>
</table>
bit plane

PSNR
15.3

bpp
.0014

Compressed size
bit planes
1 – 2

bpp
.0033

PSNR
16.8

Compressed size

CSEP 590 - Lecture 9 - Autumn 2007
bit planes
  1 – 3

bpp
  .0072

PSNR
  18.8
bit planes
1 – 4

bpp
.015

533 : 1

PSNR
20.5
bit planes
1 – 5

bpp
.035

ratio
229 : 1

PSNR
22.2
bit planes
1 – 6

bpp
.118

ratio
68 : 1

PSNR
24.8
bit planes
1 – 7

bpp
.303

ratio
26 : 1

PSNR
28.7

Compressed size
bit planes
1 – 8

bpp
.619

ratio
13 : 1

PSNR
32.9
bit planes
  1 – 9

bpp
  1.116

total ratio
  7 : 1

PSNR
  37.5
PACW

• A simple image coder based on
  – Bit-plane coding
    • Significance pass
    • Refinement pass
  – Arithmetic coding
  – Careful selection of contexts based on statistical studies

• Implemented by undergraduates Amanda Askew and Dane Barney in Summer 2003.
PACW Block Diagram

Encoder

- Image (pixels)
- Wavelet transform
- Transformed image (coefficients)
- Subtract LL Avg
- Divide into bit-planes
- Bit-plane encoding using AC
- Bit stream

Decoder

- Bit plane decoding using AC
- Recombine bit-planes
- Interpolate
- Add LL Avg
- Inverse wavelet transform
- Transformed image (approx coefficients)
- Distorted image
Arithmetic Coding in PACW

- Performed on each individual bit plane.
  - Alphabet is $\Sigma = \{0,1\}$
  - Signs are coded as needed
- Uses integer implementation with 32-bit integers. (Initialize $L = 0$, $R = 2^{32}-1$)
- Uses scaling and adaptation.
- Uses contexts based on statistical studies.
Encoding the Bit-Planes

• Code most significant bit-planes first

• Significance pass for a bit-plane
  – First code those coefficients that were insignificant in the previous bit-plane.
  – Code these in a priority order.
  – If a coefficient becomes significant then code its sign.

• Refinement pass for a bit-plane
  – Code the refinement bit for each coefficient that is significant in a previous bit-plane
Decoding

• Emulate the encoder to find the bit planes.
  – The decoder knows which bit-plane is being decoded
  – Whether it is the significant or refinement pass
  – Which coefficient is being decoded.

• Interpolate to estimate the coefficients.
Context Modeling (per bit plane)

• Significance pass contexts:
  – Contexts based on
    • Subband level
    • Number of significant neighbors
  – Sign context

• Refinement contexts
  – 1st refinement bit is always 1 so no context needed
  – 2nd refinement bit has a context
  – All other refinement bits have a context

• Context Modeling Principles
  – Bits in a given context have a probability distribution
  – Bits in different contexts have different probability distributions
Subband Level

- Image is divided into subbands until LL band (subband level 0) is less than 16x16
- Barbara image has 7 subband levels
## Statistics for Subband Levels

### Barbara (8bpp)

<table>
<thead>
<tr>
<th>Subband Level</th>
<th># significant</th>
<th># insignificant</th>
<th>% significant</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>144</td>
<td>364</td>
<td>28.3%</td>
</tr>
<tr>
<td>1</td>
<td>272</td>
<td>1048</td>
<td>20.6%</td>
</tr>
<tr>
<td>2</td>
<td>848</td>
<td>4592</td>
<td>15.6%</td>
</tr>
<tr>
<td>3</td>
<td>3134</td>
<td>23568</td>
<td>11.7%</td>
</tr>
<tr>
<td>4</td>
<td>12268</td>
<td>113886</td>
<td>9.7%</td>
</tr>
<tr>
<td>5</td>
<td>48282</td>
<td>504633</td>
<td>8.7%</td>
</tr>
<tr>
<td>6</td>
<td>190003</td>
<td>2226904</td>
<td>7.8%</td>
</tr>
</tbody>
</table>
Significant Neighbor Metric

- Count # of significant neighbors
  - children count for at most 1
  - 0,1,2,3+

Neighbors of : 

- p parent
- a spatially adjacent
- i spatially identical
- c child
## Number of Significant Neighbors

### Barbara (8bpp)

<table>
<thead>
<tr>
<th>Significant neighbors</th>
<th># significant</th>
<th># insignificant</th>
<th>% significant</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4849</td>
<td>2252468</td>
<td>.2%</td>
</tr>
<tr>
<td>1</td>
<td>13319</td>
<td>210695</td>
<td>5.9%</td>
</tr>
<tr>
<td>2</td>
<td>22276</td>
<td>104252</td>
<td>17.6%</td>
</tr>
<tr>
<td>3</td>
<td>30206</td>
<td>78899</td>
<td>27.7%</td>
</tr>
<tr>
<td>4</td>
<td>33244</td>
<td>55841</td>
<td>37.3%</td>
</tr>
<tr>
<td>5</td>
<td>27354</td>
<td>39189</td>
<td>41.1%</td>
</tr>
<tr>
<td>6</td>
<td>36482</td>
<td>44225</td>
<td>45.2%</td>
</tr>
<tr>
<td>7</td>
<td>87566</td>
<td>91760</td>
<td>48.8%</td>
</tr>
</tbody>
</table>
Refinement Bit Context Statistics

Barbara (8bpp)

<table>
<thead>
<tr>
<th></th>
<th>0’s</th>
<th>1’s</th>
<th>% 0’s</th>
</tr>
</thead>
<tbody>
<tr>
<td>2\textsuperscript{nd} Refinement Bits</td>
<td>146,293</td>
<td>100,521</td>
<td>59.3%</td>
</tr>
<tr>
<td>Other Refinement Bits</td>
<td>475,941</td>
<td>433,982</td>
<td>53.3%</td>
</tr>
<tr>
<td>Sign Bits</td>
<td>128,145</td>
<td>130,100</td>
<td>49.6%</td>
</tr>
</tbody>
</table>

- Barbara at 2bpp: 2\textsuperscript{nd} Refinement bit % 0’s = 65.8%
Context Model Details

• Significance pass contexts per bit-plane:
  – Max neighbors* num subband levels contexts
  – For Barbara: contexts for sig neighbor counts of 0 - 3 and subband levels of 0-6 = 4*7 = 28 contexts
  – Index of a context.
    • Max neighbors * subband level + num sig neighbors
    • Example num sig neighbors = 2, subband level = 3, index = 4 * 3 + 2 = 14
• Sign context
  – 1 contexts
• 2 Refinement contexts
  – 1st refinement bit is always 1 not transmitted
  – 2nd refinement bit has a context
  – all other refinement bits have a context
• Number of contexts per bit-plane for Barbara = 28 + 1 +2 = 31
Priority Queue

• Used in significance pass to decide which coefficient to code next
  – Goal code coefficients most likely to become significant
• All non-empty contexts are kept in a max heap
• Priority is determined by:
  – # sig coefficients coded / total coefficients coded
Reconstruction of Coefficients

• Coefficients are decoded to a certain number of bit planes
  – .101110XXXXX  What should X’s be?
  – .101110000... < .101110XXXXX < .101110111...
  – .101110100000  is half-way

• Handled the same as SPIHT and GTW
  – if coefficient is still insignificant, do no interpolation
  – if newly significant, add on .38 to scale
  – if significant, add on .5 to scale

\[ |A| = k \]

\[ \begin{align*}
  .A000... & \quad +.38 \quad +.5 \quad .A111... \\
  .A01100 & \quad .A100...
\end{align*} \]
Original Barbara Image
Barbara at .5 bpp (PSNR = 31.68)
Barbara at .25 bpp (PSNR = 27.75)
Barbara at .1 bpp (PSNR = 24.53)
Results

Compression of Barbara

![Graph showing PSNR (dB) vs. Bit rate (bits/pixel) for different compression methods: JPEG2000, UWIC, GTW, SPIHT, and JPEG.](image-url)
Results

Compression of Lena

![Graph showing PSNR vs Bit rate for various compression methods: PACW, GTW, SPIHT, JPEG2000, JPEG.](Diagram)
Results

Compression of RoughWall

![Graph showing compression results for different methods: UWIC, GTW, SPIHT, JPEG2000, and JPEG. The graph plots PSNR (dB) against bit rate (bits/pixel).]
PACW Notes

• PACW competitive with JPEG 2000, SPIHT-AC, and GTW.
• Developed in Java from scratch by two undergraduates, Dane Barney and Amanda Askew, in 2 months.
• Dane’s final version is slightly different than the one described here. See his senior thesis.