JPEG 2000
Descrizione ed applicazioni

Arcangelo Bruna
Advanced System Technology
Market’s requirements for still compression standard

• Application’s dependent
  – Digital Still Cameras (High / mid / low bit rate)
  – Mobile multimedia (Low / very low bit rate)

• Features requirements
  – Simple editing
  – Spatial scalability
  – Quality scalability

• JPEG – JPEG2000
Market’s requirements for video compression standard

• Application’s dependent
  – Video Cameras (High / mid / low bit rate)
  – Mobile multimedia (Low / very low bit rate)

• Features requirements
  – Simple editing
  – Spatial scalability
  – Quality scalability

• MPEG 2 (Video Cameras), MPEG4/H263 (Mobile), H264
## Current image compression standards

<table>
<thead>
<tr>
<th>Compression method</th>
<th>Input image type</th>
<th>Compression ratio</th>
<th>Controllability</th>
</tr>
</thead>
<tbody>
<tr>
<td>JBIG</td>
<td>Binary</td>
<td>(\frac{1}{2} - 10:1)</td>
<td>Lossless</td>
</tr>
<tr>
<td>JPEG</td>
<td>8, 12 bit gray/color</td>
<td>5-25:1</td>
<td>Iterate on Q-tables</td>
</tr>
<tr>
<td>JPEG-LS</td>
<td>4-12 bit</td>
<td>2-10:1</td>
<td>Lessless/near lossless</td>
</tr>
<tr>
<td>JPEG2000</td>
<td>ANY</td>
<td>Lossless-200:1</td>
<td>A lots of methods</td>
</tr>
</tbody>
</table>
## Current video compression standards

<table>
<thead>
<tr>
<th>Video Compression</th>
<th>Market</th>
<th>Video Bitrate</th>
<th>Frame-accurate editing</th>
<th>Scalability</th>
<th>Still Image Mode</th>
<th>Lossless Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPEG-1</td>
<td>Video CD authoring</td>
<td>1.0-1.5 Mbits/s @ 352x240x29.97 fps</td>
<td>No</td>
<td>Low</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MPEG-2</td>
<td>DVD authoring</td>
<td>3.0-100.0 Mbits/s @ 720x480x29.97 fps</td>
<td>No</td>
<td>Low</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MPEG-4</td>
<td>Internet Streaming</td>
<td>0.3-1.0 Mbits/s @ 352x240x29.97 fps</td>
<td>No</td>
<td>High</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>MJPEG</td>
<td>Video Production</td>
<td>10.0-80.0 Mbits/s @ 720x480x29.97 fps</td>
<td>Yes</td>
<td>Low</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>DV</td>
<td>Professional Video Production</td>
<td>25.0 Mbits/s @ 720x480x29.97 fps</td>
<td>Yes</td>
<td>Low</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MJPEG2000</td>
<td>Professional Video Production Digital Video Cameras Video/image streaming</td>
<td>2.0-50.0 Mbits/s @ 720x480x29.97 fps</td>
<td>Yes</td>
<td>High</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Conventional compression method (JPEG)
JPEG Pros and Cons

**Advantages**
- Memory efficient
- Low complexity
- Compression efficiency

**Disadvantages**
- Single resolution
- Single quality
- Difficult to control the target bit rate
- No tiling
- No region of interest
- Blocking artifacts
- Poor error resilience
JPEG 2000 flexibility

Encode choices
- tiling
- lossy/lossless
- + old paradigm choices

Decode choices
- image resolution
- image fidelity
- quantization
- region-of-interest
- fixed-size
- fixed-rate components

bit-stream
Where to use JPEG 2000?

Medical imagery
Scanners
Network imagery (WWW)
Image archival (CD-ROM)
Page description languages
Digital cameras
Compound documents
Graphic images

Satellite imagery
Printers
Pre-press imagery
Interframe video
Multi-function copiers
Facsimile
Set Top Box
etc.

briefly... everywhere!!!
JPEG 2000

The standard description
Who is it?

JPEG2000 was standardized by ISO/IEC:

- **Part 1**: the core, is now published as an International Standard
  
  ISO/IEC 15444

- **Parts 2-6**: Extensions (adds more features and sophistication to the core), Motion JPEG 2000, Conformance, Reference software (Java and C implementations), Compound image file format (document imaging, for pre-press and fax-like applications, etc.)

- **Part 7**: should have contained a Technical Report (TR) outlining guidelines for minimum support of Part 1 of the Standard

- **Parts 8-13**: JPSEC (security aspects), JPIP (interactive protocols and API), JP3D (volumetric imaging), JPWL (wireless applications), ISO Base Media File Format (common with MPEG-4)
Overview
Overview

Image samples → DC level shift → Image displacement → Tile partition → Reversible path → Irreversible path → EBCOT → Bitstream
DC level shift

• The input image data can be level shifted (optional), in order to obtain the following data range:

\[-2^{B-1} \leq x[n] < 2^{B-1}\]

where B is the image bit depth

• It is useful to obtain a zero chain in the High Pass filters in the wavelet domain
Overview

DC level shift → Image displacement → Tile partition → Reversible path

Irreversible path → EBCOT → Bitstream

Advanced System Technology
Image displacement

• The input image is mapped into a high resolution grid
• All the color components are to be mapped into the Reference Grid
• It is useful, i.e., when the ROI tool is used (described later)

$$(0,0)$$

$$(ax,ay)$$

$$(bx-1,by-1)$$
Overview

Image samples → DC level shift → Image displacement → Tile partition → Reversible path → Irreversible path → EBCOT → Bitstream
Tile partition

- The reference grid is partitioned into \textit{TILES}
- Only the \textit{tiles} comprising image’s pixels are to be considered
- Each \textit{tile} will be treated separately for further algorithms
Overview

Image samples → DC level shift → Image displacement → Tile partition → Reversible path → Irreversible path → EBCOT → Bitstream
Reversible and Irreversible paths

Reversible path

Tiles → RCT → Rev. DWT → To block coder

Irreversible path

Tiles → ICT → Irr. DWT → DZQ → To block coder
Reversible and Irreversible paths

Reversible path
- Tiles → RCT → Rev. DWT → To block coder

Irreversible path
- Tiles → ICT → Irr. DWT → DZQ → To block coder
Component transform

- The color components are decorrelated
- The $YCrCb$ color representation is used
- Two versions:
  - Reversible Color Transform (RCT)
    To be used for the lossless branch (integer)
  - Irreversible Color Transform (ICT)
    To be used for the lossy branch (floating point)
- More versions are available in the Part II
Reversible Color Transform

• It is an integer-to-integer color transform
• Used for lossless coding

Forward RCT

\[ Y = \left\lfloor \frac{1}{4} (R + 2G + B) \right\rfloor \]

\[ C_b = B - G \]

\[ C_r = R - G \]

Inverse RCT

\[ G = Y - \left\lfloor \frac{1}{4} (C_r + C_b) \right\rfloor \]

\[ R = C_r + G \]

\[ B = C_b + G \]
Irreversible Color Transform

• It is an integer-to-floating point color transform
• Used for lossy coding

\[
\begin{bmatrix}
Y \\
C_r \\
C_b
\end{bmatrix} = \begin{bmatrix}
0.299 & 0.587 & 0.114 \\
-0.169 & -0.331 & 0.5 \\
0.5 & -0.419 & -0.081
\end{bmatrix} \begin{bmatrix}
R \\
G \\
B
\end{bmatrix}
\]

\[
\begin{bmatrix}
R \\
G \\
B
\end{bmatrix} = \begin{bmatrix}
1 & 0 & 1.4021 \\
1 & -0.3441 & -0.7142 \\
1 & 1.7717 & 0
\end{bmatrix} \begin{bmatrix}
Y \\
C_r \\
C_b
\end{bmatrix}
\]

Forward ICT                  Inverse ICT
RGB vs YCC
Subsampling

- Subsampling is allowed in all the components (Y,Cr,Cb)
- Only the chroma subsampling is usually used!!!
Reversible and Irreversible paths

Reversible path
Tiles → RCT → Rev. DWT → To block coder

Irreversible path
Tiles → ICT → Irr. DWT → DZQ → To block coder
Discrete Wavelet Transform (DWT)

• Two versions
  – Reversible wavelet (integer 5/3)
  – Irreversible wavelet (Daubechies 9/7)

• Mallat schema is used
1-D Discrete Wavelet Transform

Input Signal

High Pass Filter

Detail Signal

Low Pass Filter

Average Signal
2-D Discrete Wavelet Transform

\[ f(x) \rightarrow g(n) \rightarrow \downarrow 2 \rightarrow HH \]

\[ f(x) \rightarrow h(n) \rightarrow \downarrow 2 \rightarrow HL \]

\[ f(x) \rightarrow h(n) \rightarrow \downarrow 2 \rightarrow LH \]

\[ f(x) \rightarrow h(n) \rightarrow \downarrow 2 \rightarrow LL \]
2-D Mallat schema

- Resolution 0: LL3
- Res 1 (LL2): Res 0 + LH3+HL3+HH3
- Res 2 (LL1): Res1 + LH2+HL2+HH2
- Res 3 (LL0): Res 2 + LH1+HL1+HH1
DWT example
Multilevel Wavelet Decomposition

Two dimensional – three levels wavelet decomposition
## DWT 9/7 convolution coeff

<table>
<thead>
<tr>
<th>Taps</th>
<th>Analysis Filter Bank</th>
<th>Synthesis Filter Bank</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Low pass $H_0(z)$</td>
<td>High pass $H_1(z)$</td>
</tr>
<tr>
<td>0</td>
<td>0.6029490183263579</td>
<td>1.115087052456994</td>
</tr>
<tr>
<td>±1</td>
<td>0.2668641184428723</td>
<td>-0.591271763114247</td>
</tr>
<tr>
<td>±2</td>
<td>-0.0782232665289878</td>
<td>-0.057543526228499</td>
</tr>
<tr>
<td>±3</td>
<td>-0.0168664118442874</td>
<td>0.091271763114249</td>
</tr>
<tr>
<td>±4</td>
<td>0.0267487574108097</td>
<td>-</td>
</tr>
</tbody>
</table>
Reversible and Irreversible paths

**Reversible path**
- Tiles
- RCT
- Rev. DWT
- To block coder

**Irreversible path**
- Tiles
- ICT
- Irr. DWT
- DZQ
- To block coder
Dead Zone Quantization (DZQ)

- Scalar quantization with deadzone

\[ q_i[n] = \text{sign}(x_i[n]) \cdot \left\lfloor \frac{x_i[n]}{\Delta_b} \right\rfloor \]

where \( \Delta_b \) is the quantization factor

- Different stepsize for each subband
  \( nLL, nLH, nHL, nHH, (n-1)LH, (n-1)HL, (n-1)HH, \ldots 1LH, 1HL, 1HH \)

- Trellis method is allowed in part II
Precints

- Each tile is partitioned into *Precints*
- The partitioning is similar to the *Tiles* partitioning

One component of the tile nLL
Codeblocks

• Each subband of each precinct is partitioned into **Codeblocks**

• Each **bitplane** of the codeblock is coded independently

• Every bitplane is compressed using the **EBCOT** algorithm
Overview

Image samples → DC level shift → Image displacement → Tile partition → Reversible path → Irreversible path → EBCOT → Bitstream
EBCOT

• Embedded Block Coding with Optimal Truncation

• It defines the methodology for the data arranging in the bitstream allowing the rate control with optimum quality

• It is composed by two tier
  – Tier 1: it generates the collection of encoded bits for each codeblock. Bitplane coding is used (Layers)
  – Tier 2: reorganize such data in order to obtain the better quality as possible given the compression ratio
Bitplane coding

- Code block samples
- Sample signs
- N bit sample magnitudes

Coding proceeds incrementally through bitplanes

- Plane N-1
- Plane 2
- Plane 1
- Plane 0
Packets

• Each atomic data information is inserted in an individual bitstream portion called Packet

• A packet contains the data information related to each:
  – Tile, Precint (spatial)
  – Component (Y, Cr, Cb) (color)
  – Resolution level (LL, ...) (resolution)
  – Layer (quality)
Progression order

• Indicates how the packet’s information are inserted in the bitstream

• Packets can be indexed by layer, resolution, precint, component

• 5 different progression order are defined

• The encoder choose the progression order based, e.g., on the application
Progression order

- The 5 progression orders are:
  1) layer - resolution level - component - position
  2) resolution level - layer - component - position
  3) component - position - resolution level - layer
  4) resolution level - position - component - layer
  5) position - component - resolution level - layer

E.g. the 1st progression order can be useful for low quality-full resolution decoding, while the 4th for a low resolution-high quality decoding.
Additional features

• Region of interest (ROI)

• Error resilient tools
ROI

• Allows obtaining finest quality for the most interesting image regions

• No algorithm is define to individuate the region of interest (i.e. proprietary algorithms must be used)
ROI example

The lower right is the most significative part
Errors handling

• Noise prone environments (transmission channel) can modify some bit in the bitstream

• Compressed data usually use variable length coding

• Corrupted bitstream cause the decoder to fail. All the following data are lost.

• Robust decoder must handle errors in order to be suitable for wireless multimedia applications.
Errors handling

• Two main phases for errors handling:
  – Error detection
    • It depends on both the encoder and the decoder
    • The encoder can use some tools to allow the error identification/resynchronization
    • The decoder uses such tools to identify the error and to jump to the next resynchronization point
  – Error concealment
    • It depends on the decoder
Errors handling

• JPEG is too poor in error handling
  – Resync marker is the only tool
  – When an error occurs at least all the line is lost
    (usually all the bitstream is discarded)

• JPEG 2000 has different tools for error handling
  – Part 13 in the standard (just started) is analyzing the
    wireless multimedia applications in order to provide
    specifications for robust transmission.
Error resilient tools

• The following tools are defined to increase the error detection and resynchronization
  – Resync marker
  – Segment marker
  – Precincts
  – Frequent arithmetic coder termination

• They provide robustness to the bitstream
• Error concealment algorithms are not defined in the standard. It is a research activity.
Error resilience example

Error rate: 10^-5

Original image

JPEG2000
Without error resilience

JPEG 2000
With error resilience

JPEG
Without error resilience
Error Concealment in MPEG-4

MPEG-4 example

Note the blocking artifacts
Quality comparison

- Visual tests looked for equivalent quality between JPEG 2000 and JPEG
- The results (Gormish & Marcellin) show better performances for JPEG 2000 in lower bitrate.

<table>
<thead>
<tr>
<th>JPEG 2000 bitrate</th>
<th>Eq. JPEG bitrate</th>
<th>% larger for JPEG</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.25</td>
<td>0.73</td>
<td>112%</td>
</tr>
<tr>
<td>0.50</td>
<td>0.78</td>
<td>56%</td>
</tr>
<tr>
<td>0.75</td>
<td>0.92</td>
<td>23%</td>
</tr>
<tr>
<td>1.00</td>
<td>1.13</td>
<td>13%</td>
</tr>
</tbody>
</table>
Visual quality

0.125 bpp
Visual quality

0.250 bpp
Visual quality

0.5 bpp
Visual quality

1 bpp
Visual quality (ROI)

0.25 bpp overall (without and with ROI at 0.75 bpp)
Visual quality
JPEG vs JPEG2000

Compression factor 1:60

Original
24 bpp

JPEG 2000
0.4 bpp

JPEG
0.4 bpp
Original 24 bpp
JPEG 0.4 bpp
JPEG 2000 0.4 bpp
JPEG 2000 bibliography

- www.jpeg.org - official site
- ISO-IEC 15444 docs – standard publications

- A lots of scientific papers (IEEE, SPIE, CG, ...)

Advanced System Technology