JPEG 2000:

March 1997: JTC 1.29.14 (ISO/IEC 15444-1 or ITU-T Rec. T.800

Nimrod Peleg

Nov. 2001

Why a new standard?

- Superior low bit-rate performance: below 0.25 bpp
- Lossless & lossy compression under one environment
- Large images: karger than 64k x 64k
- Single decompression architecture (current JPEG has 44 modes)
- Transmission in noisy environments
- Computer generated imagery
- Compound documents compression

A unified system

- Intended to create a new image coding system for different types of still images: bilevel, gray level, color, multi-component.
- Different characteristics: natural images, scientific, medical, remote sensing, text, rendered graphics, etc.
- different imaging models: (client/server, real-time transmission, image library archival, limited buffer and bandwidth resources, etc.

The Target

- Low bit-rate operation with rate distortion and subjective image quality performance superior to existing standards, without sacrificing performance at other points in the rate-distortion spectrum.
- Part I is an international standard since December 2000.

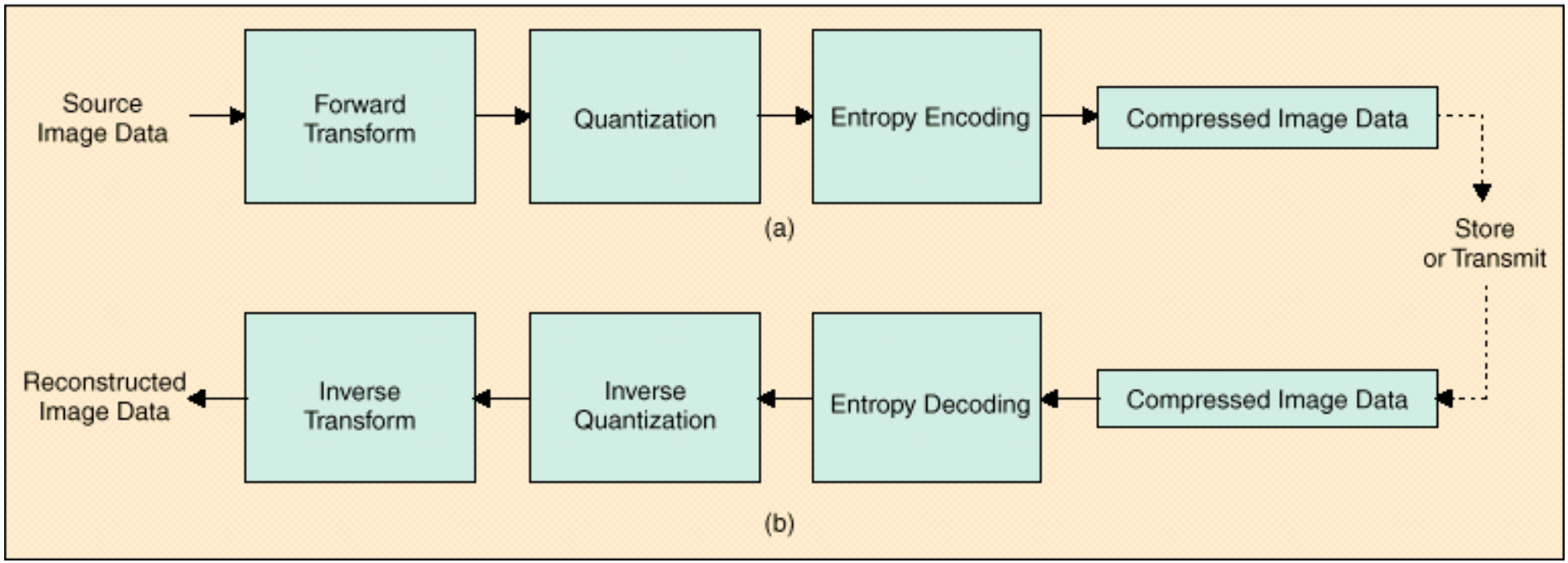
Main Features 1/2

- *Superior low bit-rate performance:* below 0.25 b/p for highly detailed gray scale images
- *Continuous-tone and bilevel compression:* various dynamic ranges (e.g., 1 to 16 bits)
- *Lossless and lossy compression:* with embedded bit stream and allow progressive lossy to lossless buildup.
- *Random access* and processing

Main Features 2/2

- *Progressive transmission by pixel accuracy and resolution*
- *Region-of-interest (ROI) coding*
- Open *architecture*
- *Robustness to bit errors*
- *Protective image security (*watermarking, labeling, stamping, or encryption)

JPEG 2000 Compression Engine



Encoder and Decoder general structure

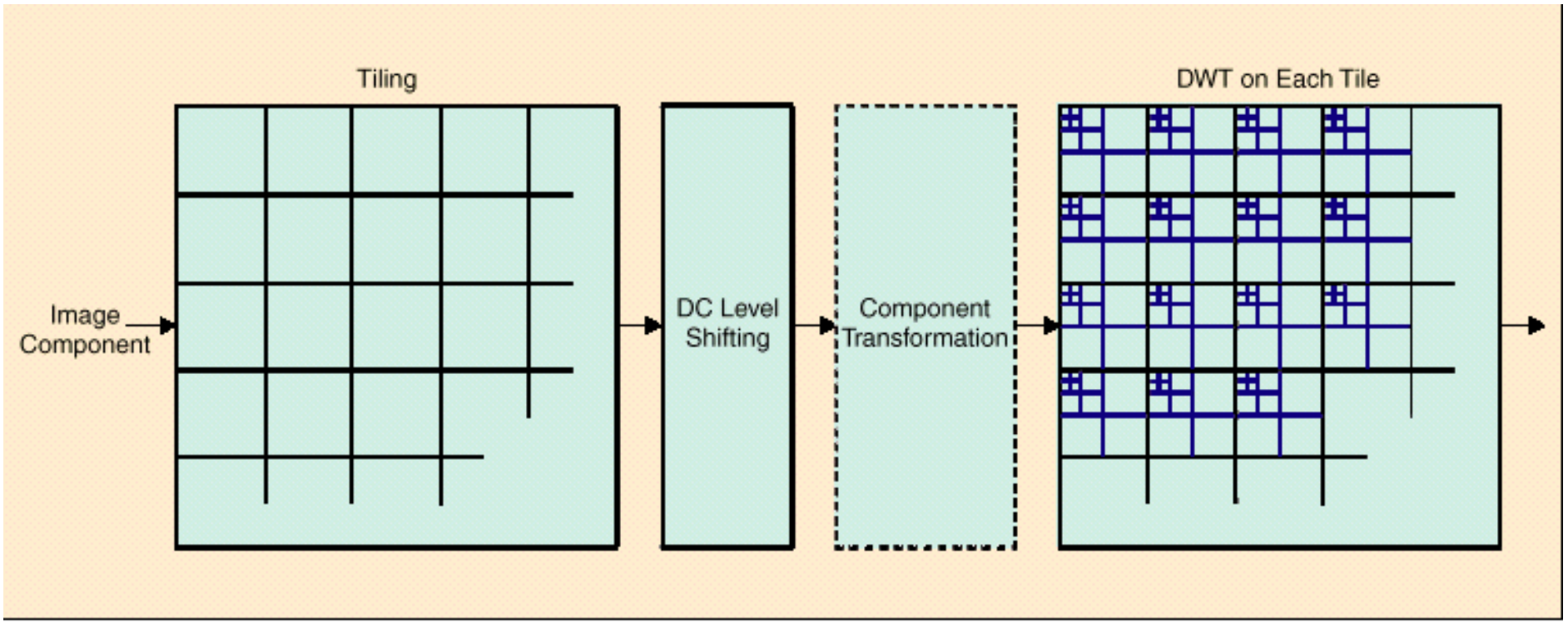
System Overview 1/2

- The source image is decomposed into components.
- The image components are (optionally) decomposed into rectangular tiles.
- A wavelet transform is applied on each tile: different resolution levels.
- The subbands of coefficients are quantized and collected into rectangular arrays of "code blocks."

System Overview 2/2

- The bit planes of the coefficients in a code block are entropy coded.
- certain regions of interest can be coded at a higher quality than the background.
- Markers are added to the bit stream to allow for error resilience.

The Pre-Processing



Tiling, dc-level shifting, color transformation (optional) and DWT of each image component.

Pre-Processing I : *Image Tiling*

- Image is partitioned into rectangular non-overlapping blocks which are compressed independently.
- All operations, including component mixing, wavelet transform, quantization and entropy coding are performed independently on the image tiles
- All tiles have exactly the same dimensions, except maybe those at the boundary of the image.
- Arbitrary tile sizes are allowed, up to and including the entire image.

Tiling Effects 1/2

(a)

(b)

(c)

(d)

Image :"ski" of size 720x576 : (a) original image, (b)-(d) reconstructed images after JPEG 2000 compression at 0.25 bpp: (b) without tiling, (c) with 128 x 128 tiling, and (d) with 64 x 64 tiling.

courtesy of Phillips Research, UK

Tiling Effects 2/2

Table 1. The Effect of Tiling on Image Quality.

Tiling / Bit Rate (b/p)	No Tiling	Tiles of Size 128 × 128	Tiles of Size 64 × 64
0.125	24.75	23.42	20.07
0.25	26.49	25.69	23.95
0.5	28.27	27.79	26.80

PSNR (in dB) for the color image "ski" (of size 720 × 576 pixels per component)

Larger tiles perform visually better than smaller tiles.
Image degradation is more severe in the case of low bit rate than the case of high bit rate.

Only 1.5 dB !

More than 4.5 dB !

Pre-Processing II: *DC Level Shifting*

- All samples of the image tile component are DC level shifted by subtracting the same quantity 2^{P-1}, where P is the component's precision.
- It actually converts an unsigned representation to a two's complement representation.
- At the decoder side, inverse DC level shifting is performed on reconstructed samples.

Pre-Processing III:

Component Transformations

- Different components need not have the same bit depths nor need to all be signed or unsigned.
- The standard supports two different component transformations:
  - one irreversible component transformation (ICT) that can be used for lossy coding
  - one reversible component transformation (RCT) that may be used for lossless or lossy coding.
- Encoding without color transformation is also possible.

The forward and the inverse ICT

$$\begin{pmatrix} Y \\ C_b \\ C_r \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.16875 & -0.33126 & 0.5 \\ 0.5 & -0.41869 & -0.08131 \end{pmatrix} \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.0 & 0 & 1.402 \\ 1.0 & -0.34413 & -0.71414 \\ 1.0 & 1.772 & 0 \end{pmatrix} \cdot \begin{pmatrix} Y \\ C_b \\ C_r \end{pmatrix}.$$

The forward and the inverse RCT

$$\begin{pmatrix} Y_r \\ V_r \\ U_r \end{pmatrix} = \begin{pmatrix} \left\lfloor \frac{R+2G+B}{4} \right\rfloor \\ R-G \\ B-G \end{pmatrix}$$

$$\begin{pmatrix} G \\ R \\ B \end{pmatrix} = \begin{pmatrix} Y_r - \left\lfloor \frac{U_r+V_r}{4} \right\rfloor \\ V_r+G \\ U_r+G \end{pmatrix}$$

The RCT is a decorrelating transformation, which is applied to the three first nents of an image. Three goals are achieved by this transformation:

- color decorrelation for efficient compression.
- color space with respect to the HVS
- ability of having lossless compression,

All three components shall have the same sampling parameters and the same bit depth.

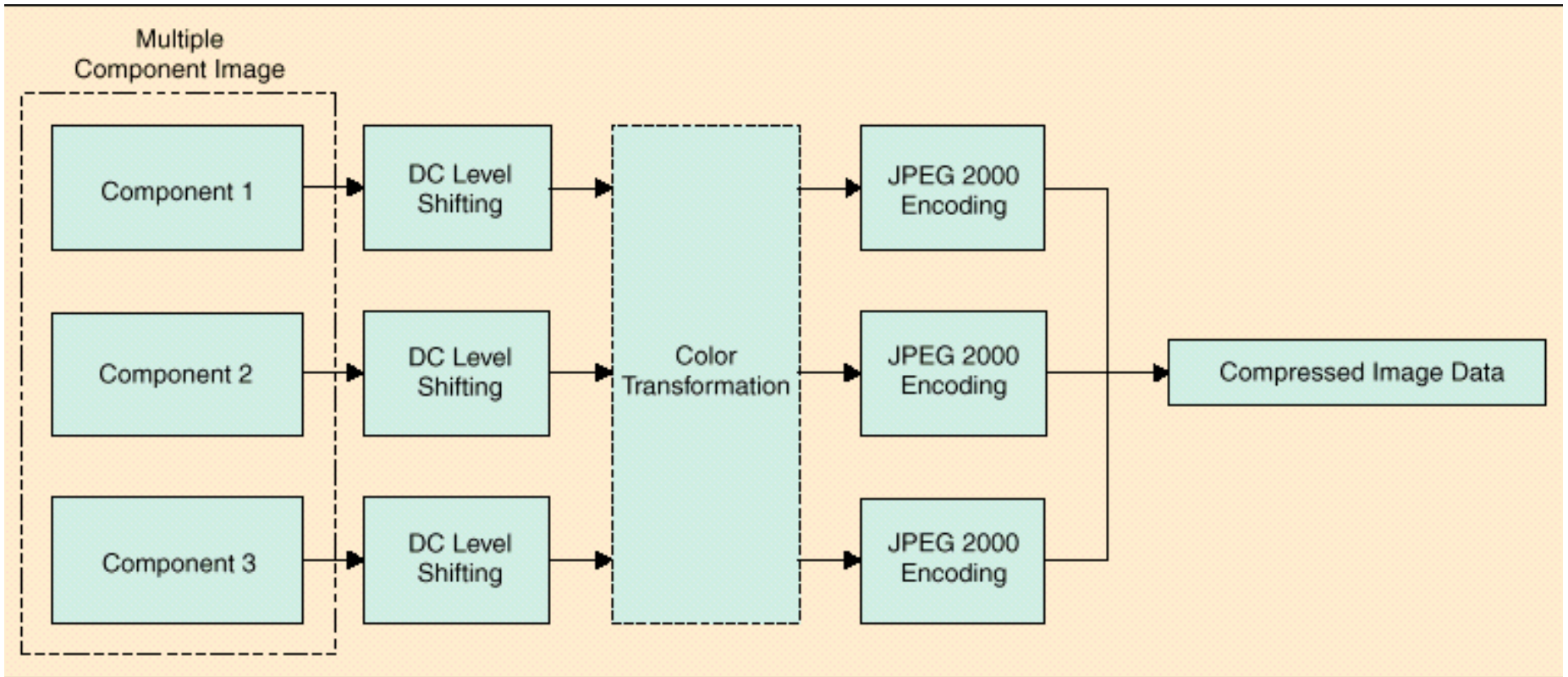
ICT/RCT Effect

Table 2. The effect of component transformation on the compression efficiency for the ski image. RCT is employed in the lossless case and ICT in the lossy case. No tiling is used.

	Without Color Transformation	With Color Transformation
Lossless compression	16.88 b/p	14.78 b/p
Lossy compression at 0.25 b/p	25.67 dB	26.49 dB

JPEG2000 does not use the RGB to YCrCb decorrelation transform (followed by sub-sampling) since the multiresolution nature of the wavelet transform may be used to achieve the same effect.

Pre-Processing full Scheme

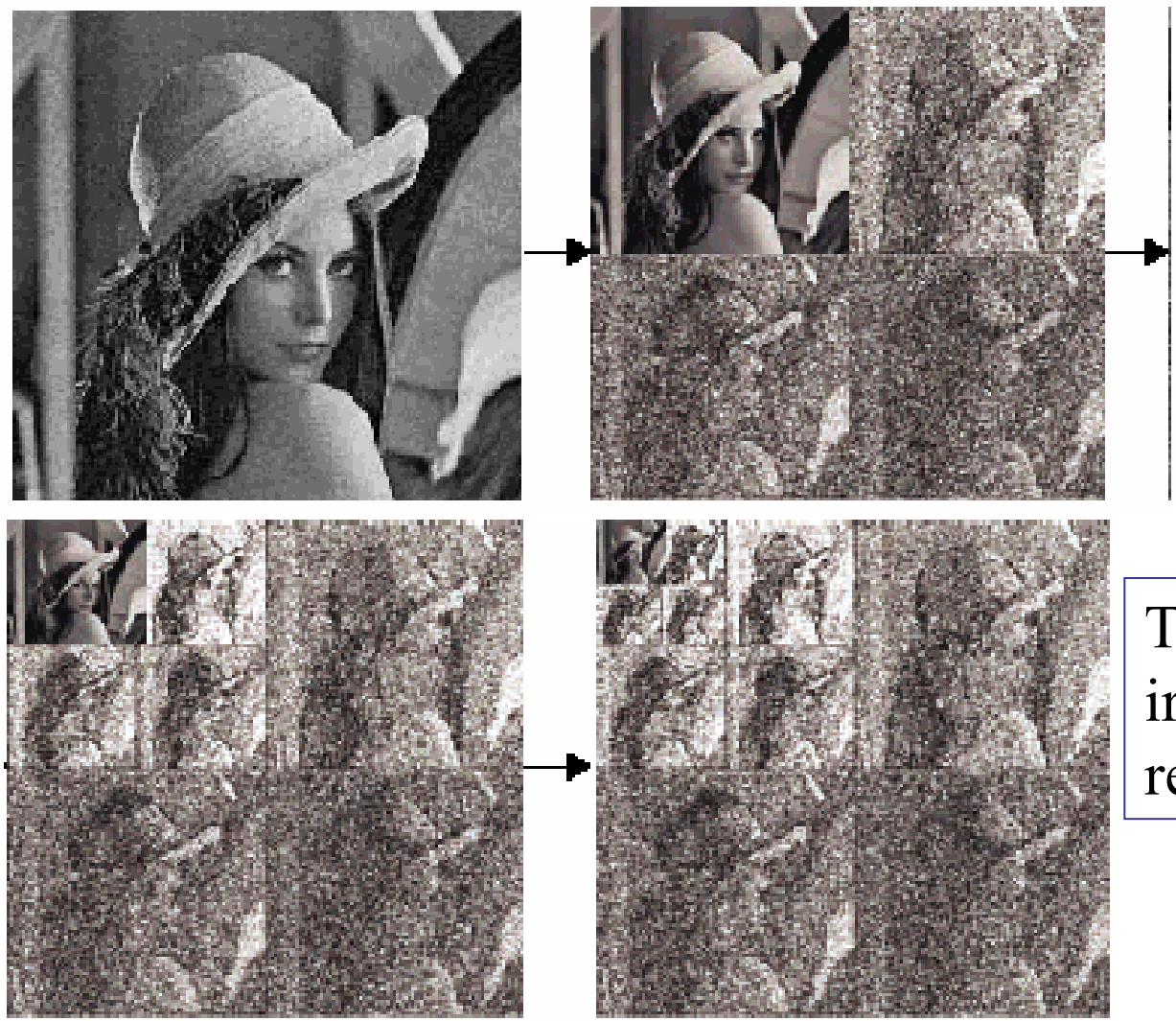


The JPEG 2000 multiple component encoder. Color transformation is optional. If employed, it can be irreversible or reversible.

The Core Processing I: DWT

- The decomposition levels contain a number of subbands, which consist of coefficients that describe the horizontal and vertical spatial frequency characteristics of the original tile component.
- In Part I of the JPEG2000 standard only power of 2 decompositions are allowed in the form of dyadic decomposition.

Three-level dyadic wavelet decomposition of the image "Lena."



The DWT can be irreversible or reversible.

The Lossy Mode

- The default irreversible transform is implemented by means of the Daubechies 9-tap/7-tap filter.
- The analysis and the corresponding synthesis filter coefficients are given the next slide.

Lossy Mode Filters

Analysis Filter Coefficients		
i	Low-Pass Filter $h_L(i)$	High-Pass Filter $h_H(i)$
0	0.6029490182363579	1.115087052456994
±1	0.2668641184428723	–0.5912717631142470
±2	–0.07822326652898785	–0.05754352622849957
±3	–0.01686411844287495	0.09127176311424948
±4	0.02674875741080976	

Synthesis Filter Coefficients		
i	Low-Pass Filter $g_L(i)$	High-Pass Filter $g_H(i)$
0	1.115087052456994	0.6029490182363579
±1	0.5912717631142470	–0.2668641184428723
±2	–0.05754352622849957	–0.07822326652898785
±3	–0.09127176311424948	0.01686411844287495
±4		0.02674875741080976

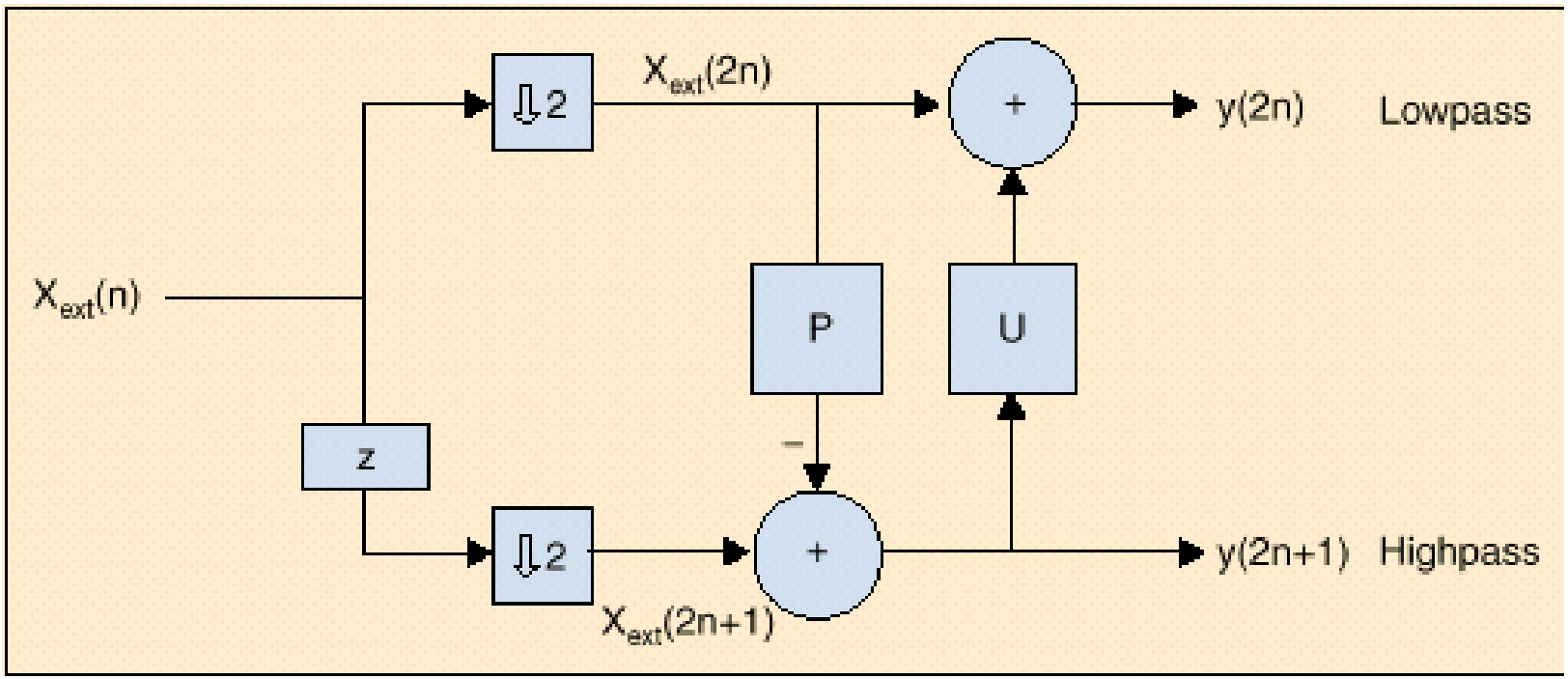
Daubechies 9/7 Analysis and Synthesis Filter Coefficients.

Lossless Mode

	Analysis Filter Coefficients		Synthesis Filter Coefficients	
i	Low-Pass Filter $h_L(i)$	High-Pass Filter $h_H(i)$	Low-Pass Filter $g_L(i)$	High-Pass Filter $g_H(i)$
0	6/8	1	1	6/8
±1	2/8	−1/2	1/2	−2/8
±2	−1/8			−1/8

The default reversible transformation is implemented by means of the Le Gall 5-tap/3-tap filter

The DWT Scheme



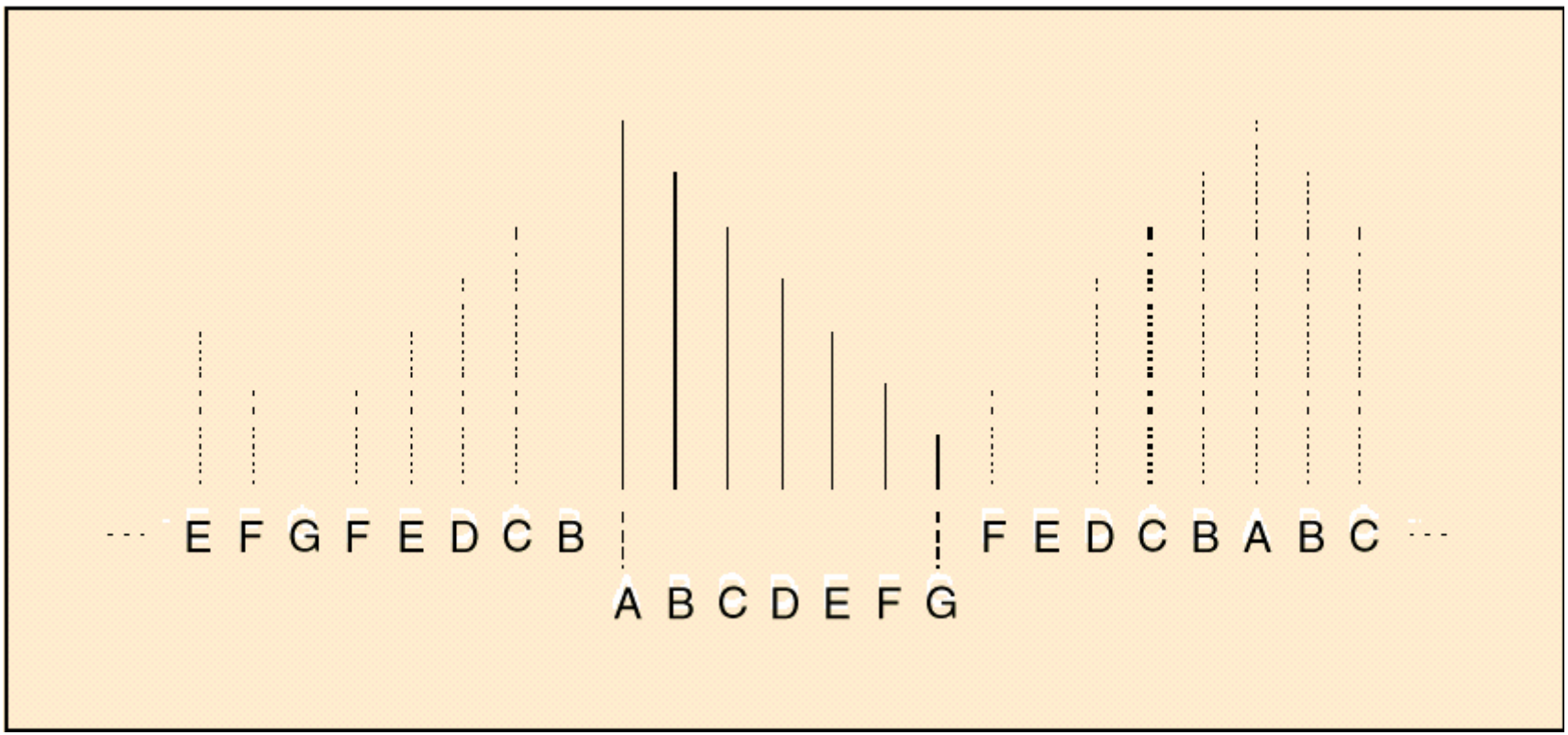
The forward (analysis) wavelet transform using lifting.
P and U stand for prediction and update, respectively.

Periodical Extension

- For both optional modes (convolution and lifting) to be implemented, the signal should first be extended periodically.
- This periodic symmetric extension is used to ensure that for the filtering operations that take place at both boundaries of the signal, one signal sample exists and spatially corresponds to each coefficient of the filter mask.
- The number of additional samples required at the boundaries of the signal is therefore filter-length dependent.

Example: *Periodic symmetric extension*

of the finite length signal "ABCDEFG."



Core Processing II: Quantization

- After transformation, all coefficients are quantized. Uniform scalar quantization with dead-zone about the origin.
- This operation is lossy, unless the quantization step is 1 and the coefficients are integers, as produced by the reversible integer 5/3 wavelet.

Quantization Formula

- Each of the transform coefficients $a_b(u,v)$ of the subband b is quantized to the value $q_b(uv)$ according to the formula:

$$q_b(u,v) = sign(a_b(u,v)) \left\lfloor \frac{|a_b(u,v)|}{\Delta_b} \right\rfloor$$

The quantization step-size Δb is represented relative to the dynamic range of subband b, i.e., JPEG 2000 supports separate quantization step-sizes for each subband ! (one per subband).

Core Processing III: Entropy Coding

- JPEG2000 uses arithmetic coding that compresses binary symbols relative to an adaptive probability model associated with each of 18 different coding contexts.
- The specific algorithm has been selected in part for compatibility reasons with the arithmetic coding engine used by the JBIG2 compression standard.

Entropy Coding: Context usage

- The arithmetic coding system uses binary symbols relative to an adaptive probability model associated with each of 18 different coding contexts.
- The restricted number of contexts allows rapid probability adaptation and decreases the cost of independently coded segments.
- The models are always reinitialized at the beginning of each code block.

Entropy Coding: Arithmetic Coding

- The arithmetic coder is always terminated at the end of each block, i.e., the end of the last sub-bit plane (useful for error resilience)

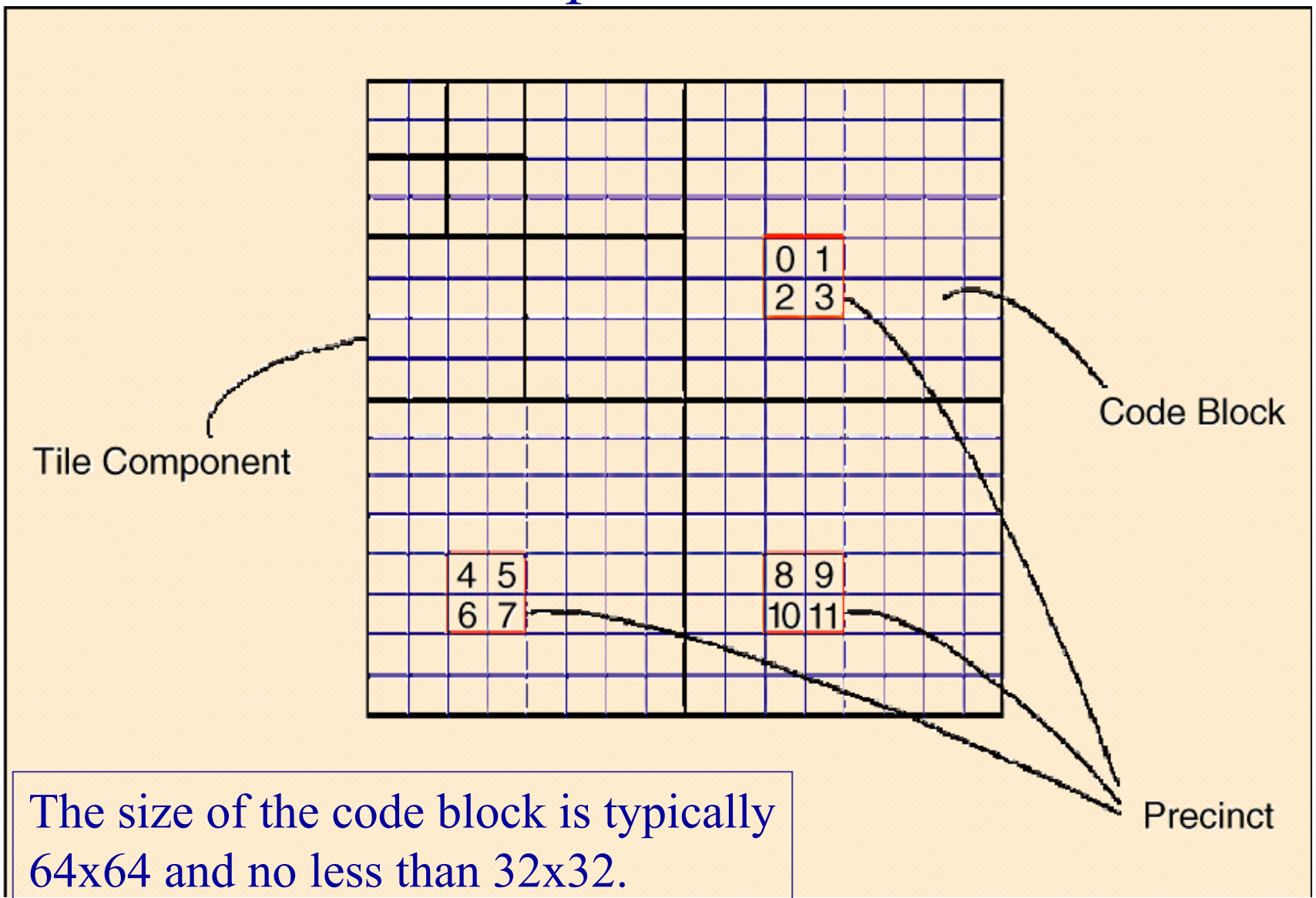
Entropy Coding: 'Lazy' Mode

- A 'lazy' coding mode is used to reduce the number of symbols that are arithmetically coded:
  - after the fourth bit plane is coded, the first and second pass are included as uncompressed data.
  - the coder is bypassed, while only the third coding pass of each bit plane employs arithmetic coding.
  - This results in significant speedup for software implementations at high bit rate.

Bit-Stream Formation

- After quantization, each subband is divided into non-overlapping rectangular blocks.
- Three spatially rectangles - one from each subband at each resolution - comprise a packet partition area (precinct).
- Each precinct is further divided into non-overlapping rectangles, called code blocks, which form the input to the entropy coder.

Partition of a tile component into code blocks and precincts.



The size of the code block is typically 64x64 and no less than 32x32.

The Code Blocks 1/2

- The code blocks are scanned in raster order, Within each subband.
- These code blocks are then coded a bit plane at a time starting with the most significant bit plane with a nonzero element, to the least significant bit plane.
- Each code block is coded independently, without reference to other blocks.

The Code Blocks 2/2

- This independent embedded block coding offers significant benefits such as:
  - spatial random access
  - efficient geometric manipulations
  - error resilience
  - parallel computations

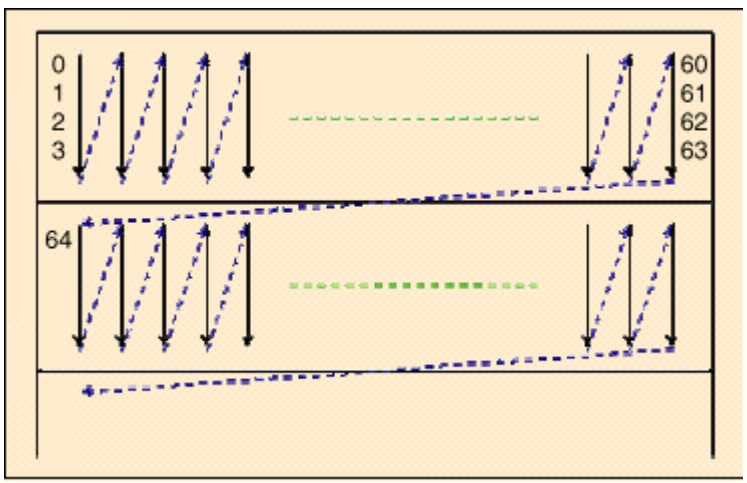
Three Coding Passes 1/2

- The individual bit planes of the coefficients in a code block are coded within 3 passes.
- Every coding pass collects context information about the bit plane data. This information is used by the arithmetic coder to generate a compressed bit stream.
- Each bit plane of a code block is scanned in a particular order, starting from the top left, the first four bits of the first column are scanned.

Three Coding Passes 2/2

- Then the first four bits of the second column, until the width of the code block is covered. Then the second four bits of the first column are scanned and so on.
- A similar vertical scan is continued for any leftover rows on the lowest code blocks in the subband.

Scan pattern of each bit plane of each code block:
selected to facilitate efficient hardware and software implementations



The Significance Propagation Pass

- Each coefficient bit in the bit plane is coded in only one of the three coding passes, (The significance propagation, the magnitude refinement, and the cleanup pass)
- For each pass, contexts are created (provided to the arithmetic coder)
- During the significance propagation pass, a bit is coded if its location is not significant, but at least one of its eight-connect neighbors is significant.

The Contexts Bins

- Nine context bins are created based on how many and which ones are significant.
- If a coefficient is significant then it is given a value of 1 for the creation of the context, otherwise it is given a value of 0.
- The significance propagation pass includes only bits of coefficients that were insignificant (the significance bit has yet to be encountered) and have a nonzero context.
  - All other coefficients are skipped.

The Magnitude Refinement Pass

- During this pass, all bits that became significant in a previous biplane are coded.
- This pass includes the bits from coefficients that are already significant.
  - except those that have just become significant in the preceding significance propagation pass.
- The context used is determined by the summation of the significance state of the horizontal, vertical, and diagonal neighbors.

The Clean-up Pass

- All the bits not encoded during the previous passes are encoded.
  - coefficients that are insignificant and had the context value of zero during the significance propagation pass.
- The cleanup pass not only uses the neighbor context, but also a run-length context.
- Run coding occurs when all four locations in the column of the scan are insignificant and each has only insignificant neighbors.

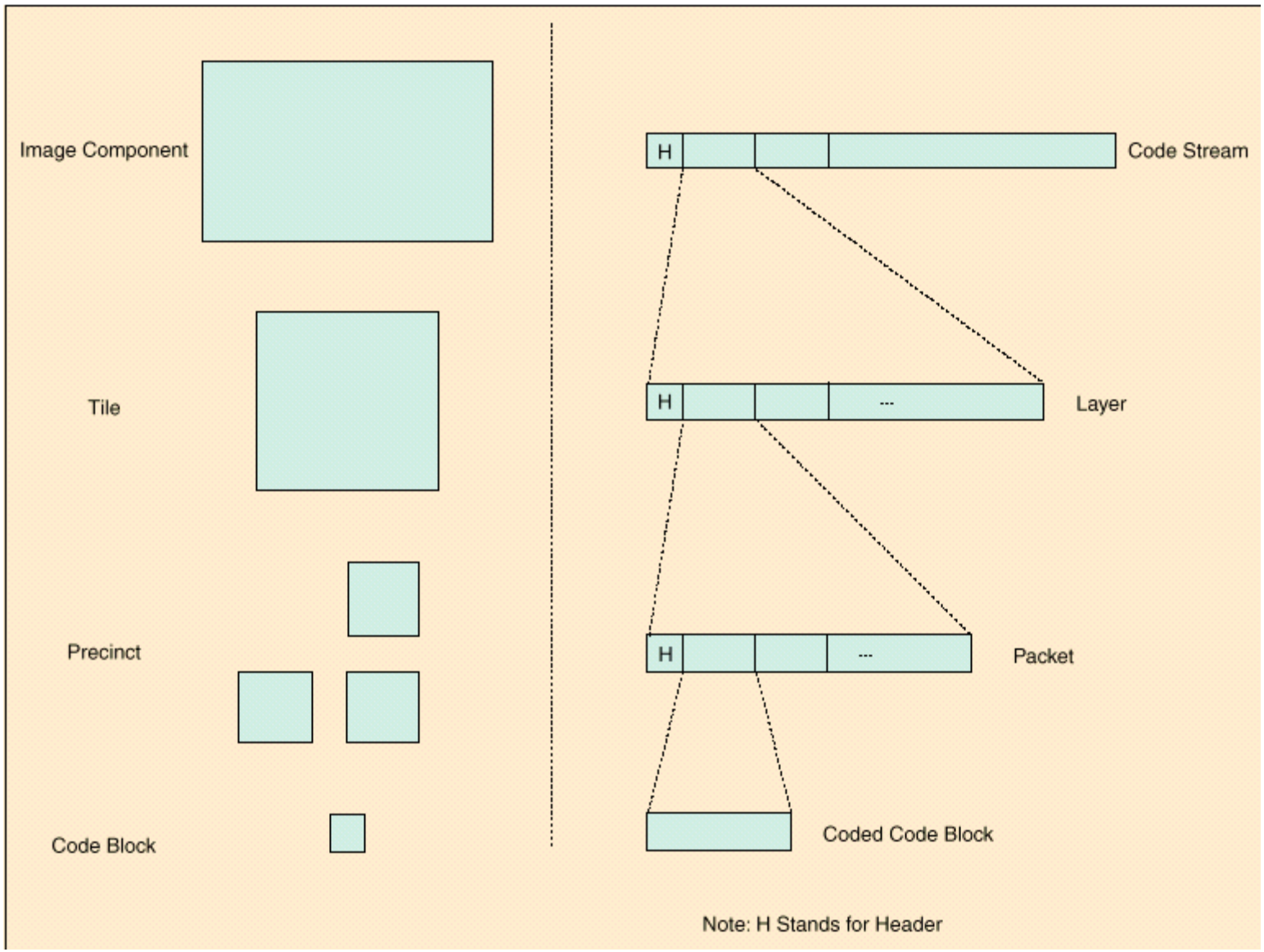
Packets and Layers

- For each code block, a separate bit stream is generated. No information from other blocks is utilized during the generation of the bit stream for a particular block.
- Rate distortion optimization is used to allocate truncation points to each code block.
- The bit stream has the property that it can be truncated to a variety of discrete lengths
  - the distortion incurred, when reconstructing from each of these truncated subsets, is estimated and denoted by the mean squared error.

Packets structure

- The compressed bit streams from each code block in a precinct comprise the body of a packet.
- A collection of packets, one from each precinct of each resolution level, comprises the layer
- A packet could be interpreted as one quality increment for one resolution level at one spatial location, since precincts correspond roughly to spatial locations.
- Similarly, a layer could be interpreted as one quality increment for the entire full resolution
- Each layer successively and monotonically improves the image quality

Conceptual correspondence between the spatial and the bit stream representations.



Progressive Modes

- There are four types of progression:

resolution, quality, spatial and component.

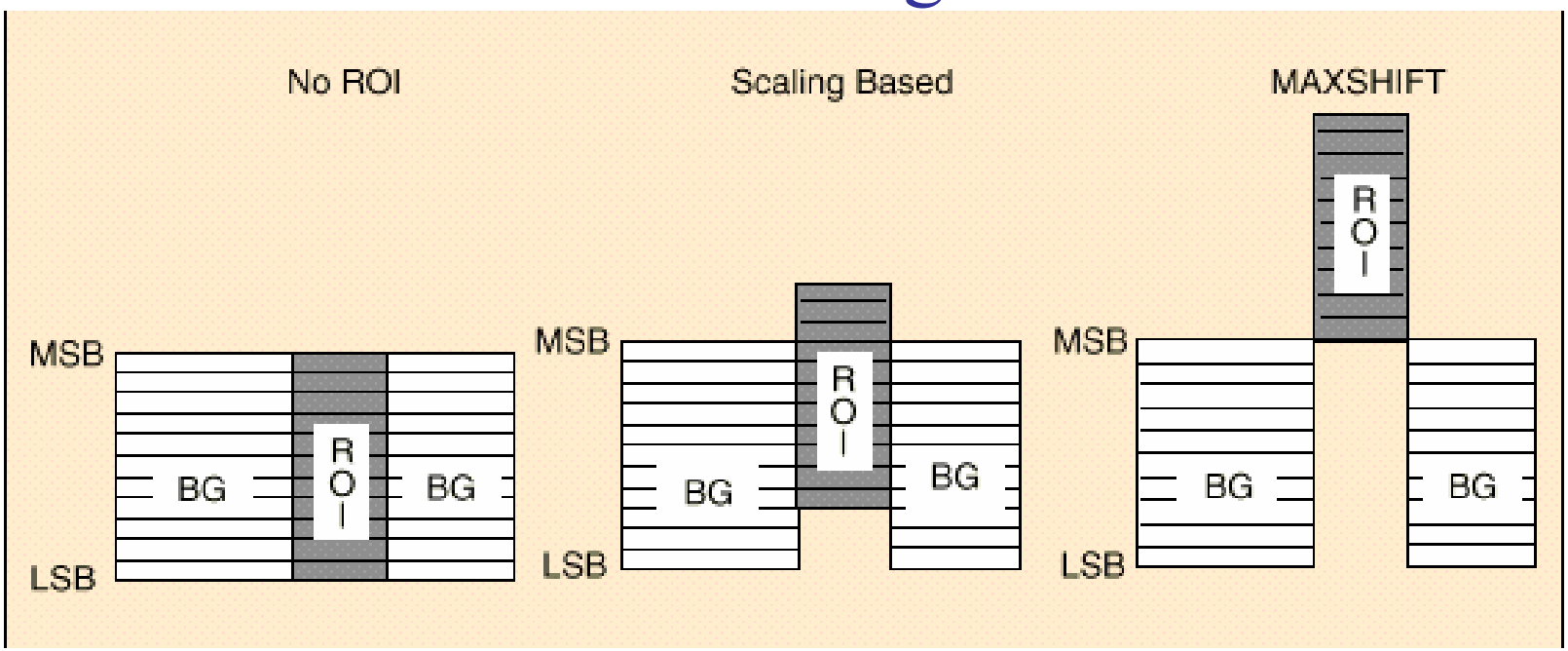
- Once the entire image has been compressed, a post-processing operation passes over all the compressed code blocks. This operation determines the extent to which each code block's should be truncated to achieve a target bitrate or distortion.
- The first, lowest layer (of lowest quality), is formed from the optimally truncated code block bit streams.
- Each subsequent layer is formed by optimally truncating the code block bit streams to achieve successively higher target bit rates.

Region of Interest (ROI)

- When certain parts of the image are of higher importance than others these regions need to be encoded at higher quality than the background.
- The ROI coding scheme in Part I of the standard is based on the called MAXSHIFT*.
  - * C.A. Christopoulos, J. Askelof, and M. Larsson, "Efficient methods for encoding regions of interest in the upcoming JPEG 2000 still image coding standard," *IEEE Signal Processing Lett.*, vol. 7, pp. 247-249, Sept. 2000.

ROI: *MAXSHIFT*

- The principle of the general ROI scaling-based method is to scale (shift) coefficients so that the bits associated with the ROI are placed in higher bit planes than the bits associated with the background.



MAXSHIFT - con't

- During the embedded coding process, the most significant ROI bit planes are placed in the bit stream before any background bit planes.
- Depending on the scaling value, some bits of the ROI coefficients might be encoded together with non-ROI coefficients
  - the ROI will be decoded, or refined, before the rest of the image.
  - a full decoding of the bit stream results in a reconstruction of the whole image with the highest fidelity available.
  - If the bit stream is truncated, or the encoding process is terminated before the whole image is fully encoded, the ROI will be of higher fidelity than the rest of the image.