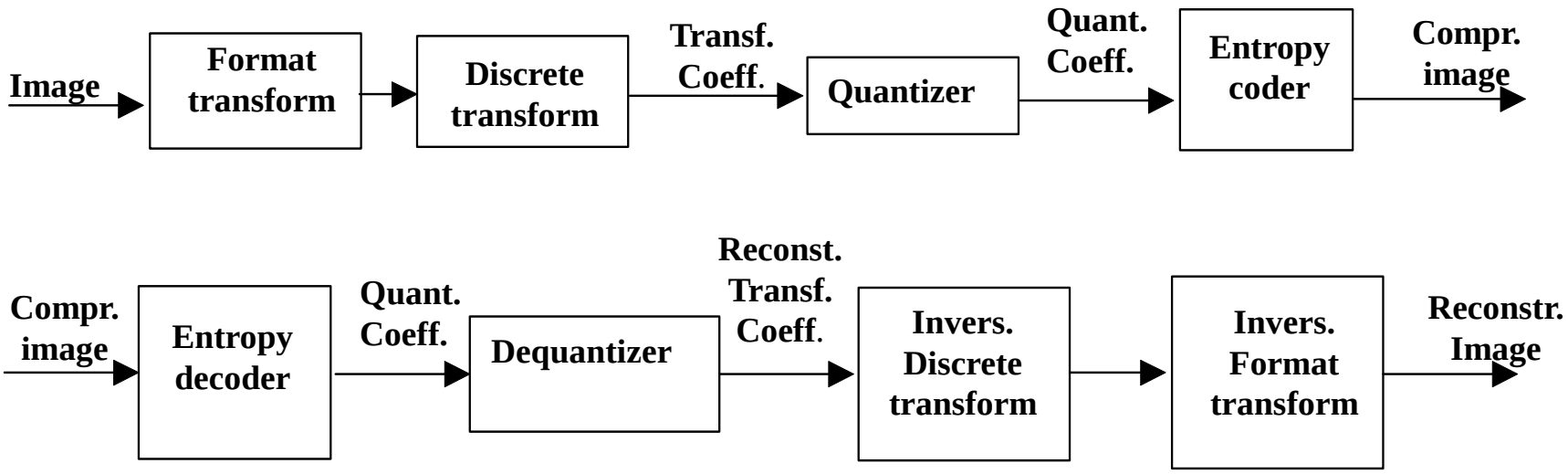
Coding of halftone images. JPEG Standard



Transform-based coder and decoder

JPEG

JPEG is an international standard for colour image compression. JPEG is an acronym for "Joint Photographic Experts Group".

This group has been working under the auspices of ISO, ITU and International Electrotechnical Comission (IEC) – for the purpose of developing a standard for colour image compression.

RGB to YUV transform

RGB to YUV transform

The JPEG compression algorithm is applied to colour images represented in a so-called RGB format. This format implies that each pixel of the colour image is represented by 3 bytes which determine shades of red (R), green (G) and blue (B) colour, respectively.

We consider 3 independent arrays to describe the colours.

Components R, G, and B are highly correlated but they are processed independently.

Colour spaces or colour coordinate systems in which one component is luminance and the other two are components are related to hue and saturation are called luminance-chrominance representations.

RGB to YUV format

One of the luminance-chrominance representations is called YUV format and the other is called YCbCr format.

We can convert RGB format to YUV format and YCbCr format using the following linear transforms, respectively

$$Y = 0.299R + 0.587G + 0.114B$$

$$V = (R - Y)0.7132$$

$$U = (B - Y)0.5643$$

$$Y = 0.299R + 0.587G + 0.114B$$

$$Cb = (B - Y)0.5643 + 128$$

$$Cr = (R - Y)0.7132 + 128$$

YUV to RGB transform

The inverse transform can be described as follows

$$G = Y - 0.714V - 0.334U$$

$$R = Y + 1.402V$$

$$B = Y + 1.772U,$$

$$G = Y - 0.714(V - 128) - 0.334(U - 128)$$

$$R = Y + 1.402(V - 128)$$

$$B = Y + 1.772(U - 128)$$

RGB to YUV transform

The components Y,U,V (Y,Cb,Cr) are almost not correlated. Moreover the most important information is concentrated in the luminance component. Thus we do not lose much information if we decimate chrominance components.

Usually U and V components are decimated with factor 2.

4 neighbouring pixels which form the square of size 2x2 are described by 4 values of component Y, one value of component U, and one value of component V.

Each chrominance value is computed as the rounded-off arithmetic mean of the corresponding 4 pixel values belonging to the considered square. As a result we obtain YUV 4:1:1 standard video format. Using it we spend 6 bytes for each 2x2 square instead 12 bytes spent by RGB format.

JPEG

JPEG standard is based on DCT coding technique. Component Y and decimated components U and V are processed by blocks. Each block of Y contains 8x8 and the corresponding blocks of U and V contain 4x4 pixels.

The 2-D DCT is applied to each block of pixels.

The 1-D DCT for sequence of length 8 is given by the formula:

$$X(k) = \frac{c(k)}{2} \sum_{n=0}^{7} x(n) \cos\left(\frac{(2n+1)k\pi}{16} \right), \quad k = 0,1,...,7$$

where

$$c(k) = \begin{cases} 1/\sqrt{2}, & \mathrm{k} = 0 \\ 1, & \mathrm{k} \neq 0 \end{cases}.$$

JPEG

The inverse transform can be written as

$$x(n) = \sum_{k=0}^{7} X(k) \frac{c(k)}{2} \cos\left(\frac{(2n+1)k\pi}{16} \right).$$

The coefficient that scales the constant basis function (k=0) is called the DC coefficients. The other coefficients are called AC coefficients (direct- and alternating-current).

Since DCT is a separable transform the 2-D DCT for blocks 8x8 can be performed by first applying 1-D transform to rows of each 8x8 block and then applying 1-D transform to the columns of the resulting block, that is,

$$X(k,l) = \frac{c(k)}{2} \sum_{i=0}^{7} \left[\frac{c(l)}{2} \sum_{j=0}^{7} x(i,j) \cos\left(\frac{(2j+1)l\pi}{16} \right) \right] \cos\left(\frac{(2i+1)k\pi}{16} \right),$$

$x(i,j)$ is the pixel of Y(U or V), $X(k,l)$ is transform coefficient.

JPEG

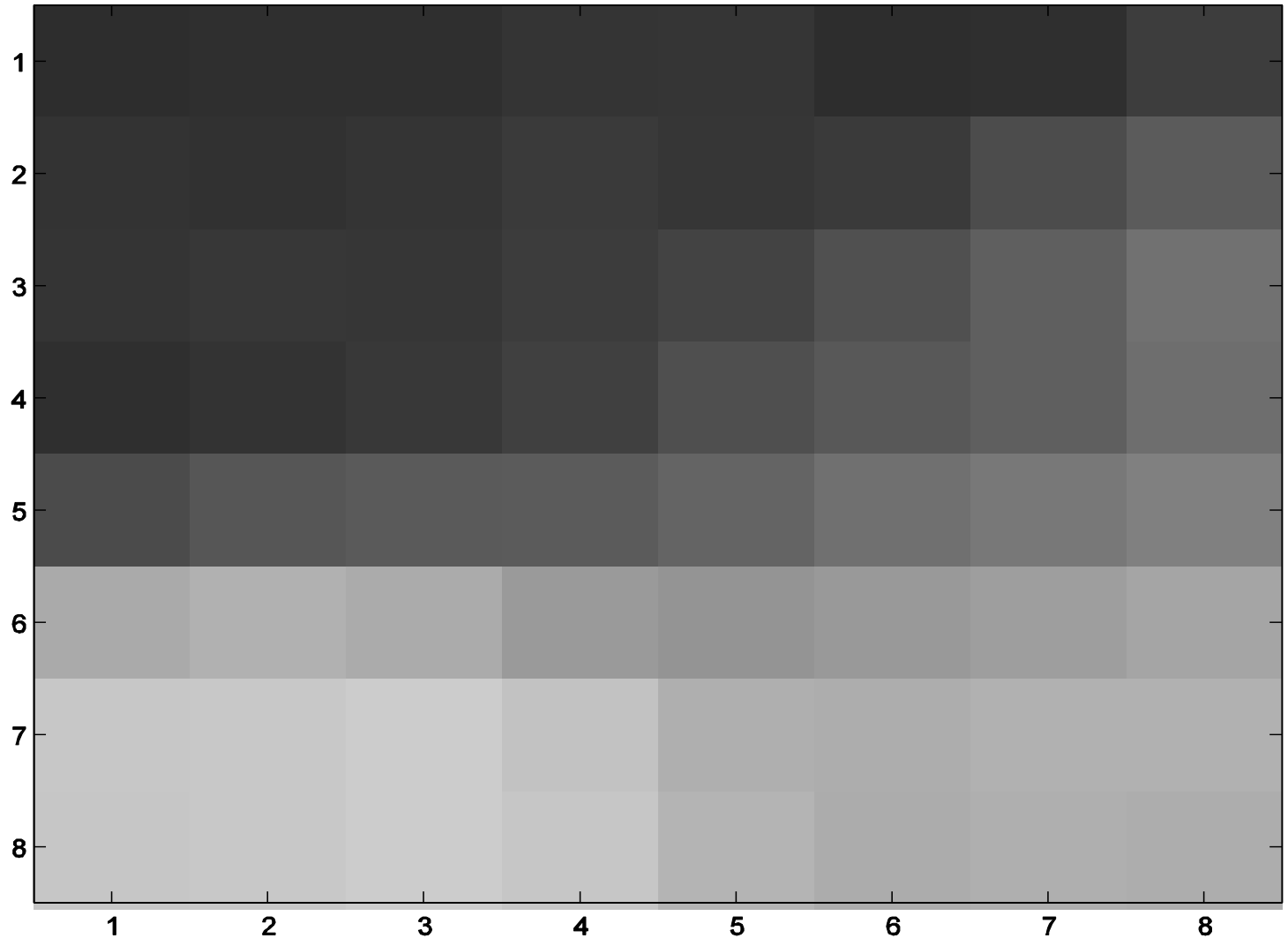
2-D DCT is equivalent to decomposition of the original image block using a set of 64 2-D cosine basis functions.

These basis functions are created by multiplying a horizontally oriented set of 1-D 8-point basis functions by a vertically oriented set of the same functions.

The only constant 2-D basis function is in the upper left corner of the array (DC coefficient).

The horizontal DCT frequency of the basis function increases from left to right and the vertical DCT frequency of basis function increases from top to bottom (AC coefficients).

8x8 block of Y component



8x8 block of Y component

46.97	52.09	53.86	48.98	76.70	171.44	200.99	199.87
49.88	50.92	56.10	52.89	87.99	178.76	201.58	201.66
49.71	53.81	55.40	57.68	91.11	172.59	205.25	205.44
53.38	59.29	61.30	65.96	92.30	156.36	196.12	199.69
54.20	55.59	68.70	80.97	101.80	149.49	176.20	181.18
47.60	59.67	82.96	90.15	113.08	154.03	174.67	173.57
48.68	77.84	96.26	97.71	121.28	159.94	178.80	176.65
62.17	93.98	114.61	111.13	129.68	166.86	178.02	174.68

DCT coefficients for 8x8 block

907.28	-416.90	90.49	18.43	-71.09	11.31	15.25	-8.62
-42.53	-67.13	68.73	27.93	-13.67	8.20	8.37	-9.23
24.55	8.72	-10.39	-1.25	-16.65	-5.53	5.02	-2.22
-24.60	4.63	-7.93	-1.39	5.71	3.56	1.82	0.66
1.36	11.60	5.38	-4.75	2.28	1.61	-2.21	-1.43
-2.06	-6.70	7.42	-4.08	-4.01	-1.03	-5.17	0.31
0.56	-0.41	-1.41	-0.01	-0.51	1.77	0.25	0.16
-2.14	-0.53	-0.13	0.76	1.43	-0.14	-0.61	-1.32

Uniformly scalar quantized 8x8 block of coefficients

113	-35	8	2	-6	1	1	-1
-4	-6	6	2	-1	1	1	-1
2	1	-1	0	-1	0	0	0
-2	0	-1	0	0	0	0	0
0	1	0	0	0	0	0	0
0	-1	1	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

JPEG

The obtained DCT coefficients are quantized by the uniform scalar quantizer. The quantization is implemented as rounding-off of the DCT coefficients divided by a quantization step.

The values of steps are set individually for each DCT coefficient, using criteria based on visibility of the basis functions. Thus the quantized coefficient is

$$Z(k,l) = round(Y(k,l)/q(k,l)) = \lfloor (Y(k,l) \pm q(k,l)/2)/q(k,l) \rfloor,$$

$$k,l = 0,1,...,7,$$

where $q(k,l)$ is the (k,l)th entry of the quantization matrix Q of size 8x8. JPEG uses two different matrices.

JPEG

The first matrix is used to quantize the luminance component (Y) and has the form

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

JPEG

The second matrix is used to quantize the chrominance components (U,V) and has the form

$$Q = \begin{pmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{pmatrix}$$

JPEG. Variable length coding

The quantized DCT coefficients are coded by VLC. The coding procedure consists in two steps.

DC coefficient is DPCM coded, using the first order predictor that is the DC value of the previous 8x8 block from the same component. AC coefficients are coded by run-length coder.

At the second step the obtained values are coded by the Huffman coder.

The difference $DC_i - DC_{i-1}$ takes values from the range [-2047,2047]. This range is split into 12 categories, where the i th category includes the differences with length of their binary representation equal to i bits.

JPEG. Variable length coding

Category	Numbers
0	0
1	-1,1
2	-3,-2,2,3
3	-7,...,-4,4,...7
4	-15,...,-8,8,...15
5	-31,...-16,16,...31
6	-63,...-32,32,...63
7	-127,...-64,64,...127
8	-255,...-128,128,...255
9	-511,...-256,256,...511
10	-1023,..-.512,512,...1023
11	-2047,...-1024,1024,...2047
12	-4095,...-2048,2048,...4095
13	-8191,...-4096,4096,...8191
14	-16383,...-8192,8192,...16383
15	-32767,...-16384,16384,...32767
16	32768

JPEG. Variable length coding

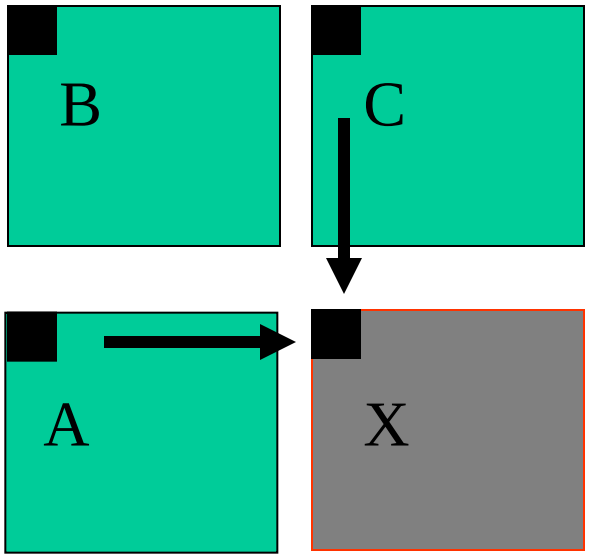
Each DC coefficient is described by a pair (category,amplitude).

If $DC_i - DC_{i-1} \geq 0$ then the amplitude is the binary representation of this difference of length equal to the category.

If $DC_i - DC_{i-1} < 0$ then the amplitude is the codeword of the complement binary code for the absolute value of the difference, which has the length equal to the category.

The category is then coded by the Huffman coder.

DC prediction



IF

$$|QDC_A - QDC_B| < |QDC_B - QDC_C|$$

$$QDC_p = QDC_C$$

ELSE

$$QDC_p = QDC_A$$

JPEG. DC coding. Example

Let $DC_{i-1} = 98$ and $DC_i = 113$ then the difference

$$DC_i - DC_{i-1} = 113 - 98 = 15.$$

The value 15 belongs to the category 4.

The binary representation of 15 is 1111.

The difference is represented as (4,1111).

If the codeword of the Huffman code for 4 is 110 then we obtain the codeword 1101111.

The decoder first processes the category value (in our case 4) and then the next 4 bits correspond to the value $DC_i - DC_{i-1}$.

Using categories simplify the Huffman code.

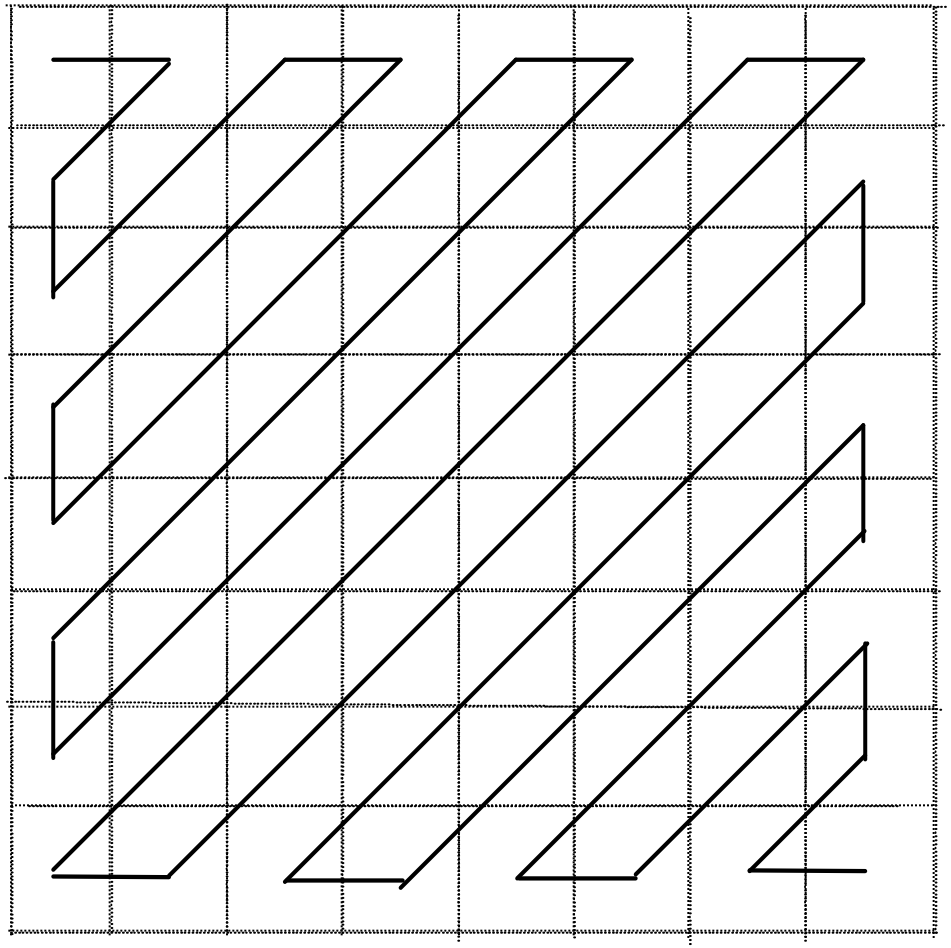
JPEG. Variable length coding

AC coefficients can take values from the range [-1023,1023].

After quantization many of these coefficients become zero. It is necessary to encode only a small number of non-zero coefficients simply indicating before their positions. To do this efficiently the 2-D array of DCT coefficients is rearranged into a 1-D linear array by scanning in the zigzag order.

This zigzag index sequence creates a 1-D vector of coefficients, where the lower DCT frequencies tend to be at lower indices. When coefficients are ordered in this fashion the probability of coefficients being zero is an approximately monotonic increasing function of the index.

JPEG. AC coding



JPEG. AC coding.

The run-length coder generates a codeword

((run-length,category),amplitude),

where run-length is the length of zero run followed by the given non-zero coefficient, amplitude is the value of this non-zero coefficient and category is the number of bits needed to represent the amplitude.

The pair (run-length,category) is coded by 2-D Huffman code and the amplitude is coded as in the case of DC coefficients and added to the codeword.

AC coding. Example.

-35, -4, 2,-6,8,2,6,1,-2,0,0,-1,2,-6,1,-1,0,-1,1,0,0, -1,0,0,

-1,1,1,-1,1,0,0,0,1,0,0,0,0,0,0,0,0,-1

((0,6),-35), (0,3),-4),((0,2),2),((0,3),-6),((0,4),8),((0,2),2)),

((0,3),6),((0,1),1),((0,2),-2),((2,1),-1), ((0,2),2), ((0,3),-6),

((0,1),1), ((0,1),-1),((1,1),-1),((0,1),1),((2,1),-1),((2,1),-1),

((0,1),1),((0,1),1),((0,1),-1), ((0,1),1),((3,1),1),((8,1),-1)

((0,6),011100),((0,3),011),((0,2),10),((0,3),001),((0,4),1000),

((0,2),10),((0,3),110),((0,1),1),((0,2),01),((2,1),0),((0,2),10),

….

JPEG. AC coding

There are two special cases when we encode the AC coefficients:

- After a non-zero coefficient all other AC coefficients are zero. In this case the special symbol (EOB) is transmitted which codes end-of-block condition
- A pair (run-length,category) appeared which is not included in the table of the Huffman code. In this case a special codeword called escape-code followed by the uniform codes for run-length and category is transmitted.

Reconstructed image

JPEG

The main shortcoming of JPEG standard is the so-called “blocking-artifacts” which appear in the reconstructed image.

These specific distortions arise since transform and quantization are applied to blocks of image.

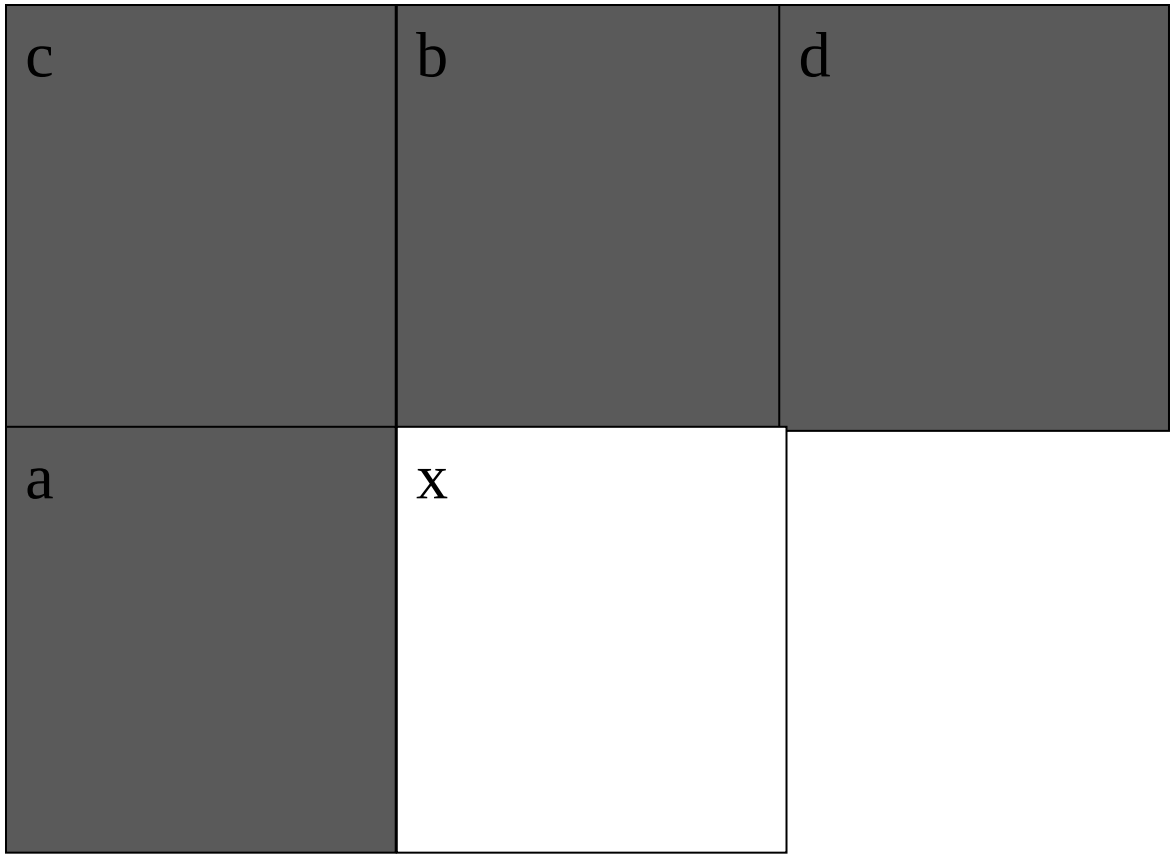
JPEG-LS

LOw COmplexity LOssless COding

Main steps:

- Prediction
- Context modeling
- Entropy coding

JPEG-LS



JPEG-LS

- Prediction is calculated as

$$\hat{x} = \begin{cases} \min(a,b) \text{ if } c \geq \max(a,b) \\ \max(a,b) \text{ if } c \leq \min(a,b) \\ a+b-c \end{cases}$$

- For context modeling gradients

$$D_1 = d-b \qquad D_2 = b-c \qquad D_3 = c-a$$

are used. Gradients are quantized into 9 cells (-4,-3,…,0,…,3,4). It gives 9x9x9=729 contexts. The number of contexts is reduced to 365 contexts by merging symmetric contexts.

JPEG-LS

- For each context we estimate bias and parameter k which implicitly characterizes exponential decay of the conditional distribution. (It is assumed that prediction error is modeled by 2-sided geometrical distribution with 0 mean)
- The Golomb-Rice code with parameter is used for encoding prediction error

For example, $T = 2^k$ e=14, T=4,

unary $\left(\left\lfloor \frac{14}{4} \right\rfloor + 1\right)$ 0001 (prefix)

e mod T 10 (suffix) codeword 000110

- $T = 2^2$ 0100000000000001 is encoded as 10100111 in run mode

Polyphase representation

$$y(0) = x(0)h(0)$$

$$y(1) = x(1)h(0) + x(0)h(1)$$

$$y(2) = x(2)h(0) + x(1)h(1) + x(0)h(2)$$

$$y(3) = x(3)h(0) + x(2)h(1) + x(1)h(2) + x(0)h(3)$$

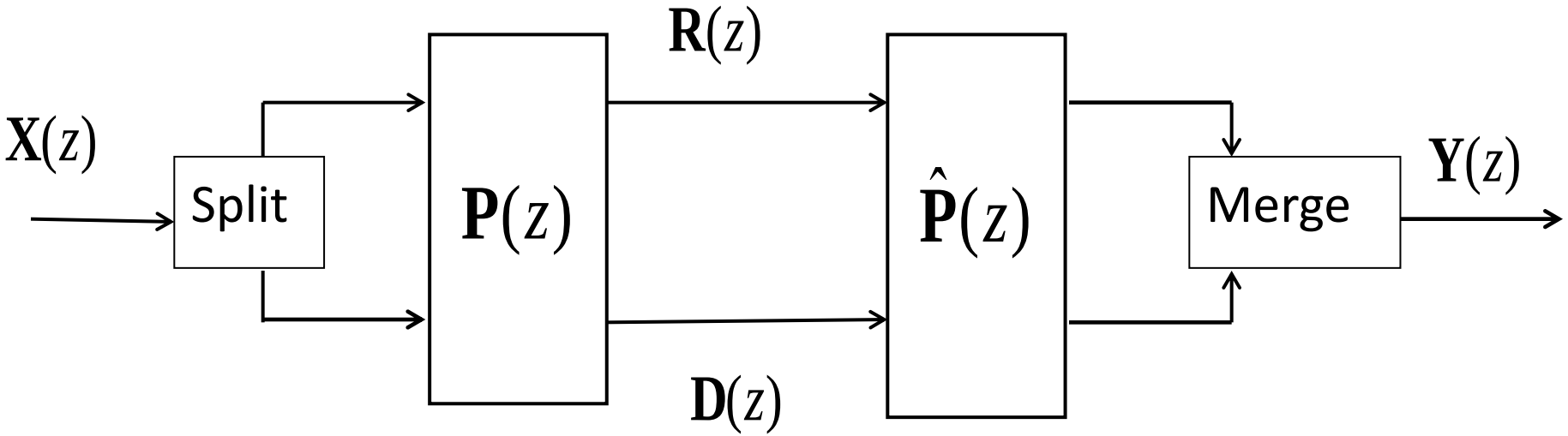
$$y(4) = x(4)h(0) + x(3)h(1) + x(2)h(2) + x(1)h(3)$$

$x(0), x(1), x(2), x(3), x(4), \ldots$ Split → $x(0), x(2), x(4), \ldots$

$x(1), x(3), \ldots$

$$Y_e(z) = H_e(z)X_e(z) + z^{-1}H_o(z)X_o(z)$$

Polyphase representation



Polyphase representation

$$x(0) = y(0)g(0)$$

$$x(1) = 0g(0) + y(0)g(1)$$

$$x(2) = y(2)g(0) + 0g(1) + y(0)g(2)$$

$$x(3) = 0g(0) + y(2)g(1) + 0g(2) + y(2)g(3)$$

....

Polyphase representation

$$\begin{pmatrix} R(z) \\ D(z) \end{pmatrix} = P(z) \begin{pmatrix} X_e(z) \\ z^{-1} X_o(z) \end{pmatrix}$$

$$P(z) = \begin{pmatrix} H_{Le}(z) & H_{Lo}(z) \\ H_{He}(z) & H_{Ho}(z) \end{pmatrix}$$

Polyphase matrix

$$Y_e(z) = G_e(z) X_e(z)$$

$$zY_o(z) = G_o(z) X_e(z)$$

Upsampled zeros are odd samples of X(z)
z is delay between odd and even samples

$$\begin{pmatrix} Y_e(z) \\ zY_o(z) \end{pmatrix} = \hat{P}(z) \begin{pmatrix} R(z) \\ D(z) \end{pmatrix}$$

$$\hat{P}(z) = \begin{pmatrix} G_{Le}(z) & G_{He}(z) \\ G_{Lo}(z) & G_{Ho}(z) \end{pmatrix}$$

$$P(z^{-1}) \hat{P}(z) = I$$

Polyphase representation

$$H_L(z) = -\frac{1}{8}z^{-2} + \frac{1}{4}z^{-1} + \frac{3}{4} + \frac{1}{4}z - \frac{1}{8}z^2$$

$$H_H(z) = \frac{1}{4}z^{-2} - \frac{1}{2}z^{-1} + \frac{1}{4}$$

$$H_{Le}(z) = -\frac{1}{8}z^{-1} + \frac{3}{4} - \frac{1}{8}z^{1} \qquad H_{Lo}(z) = \frac{1}{4} + \frac{1}{4}z$$

$$H_{He}(z) = \frac{1}{4}z^{-1} + \frac{1}{4} \qquad H_{Ho}(z) = -\frac{1}{2}$$

Lifting scheme

$$P(z)=\begin{pmatrix} H_{Le}(z) & H_{Lo}(z) \\ H_{He}(z) & H_{Ho}(z) \end{pmatrix}=\begin{pmatrix} C_1 & 0 \\ 0 & C_2 \end{pmatrix}\prod_{i=1}^{m}\begin{pmatrix} 1 & S_i(z) \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ T_i(z) & 1 \end{pmatrix}$$

$$P(z)=\begin{pmatrix} 1 & 0 \\ 0 & -1/2 \end{pmatrix}\begin{pmatrix} 1 & 1/4+(1/4)z \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ (-1/2)z^{-1}-1/2 & 1 \end{pmatrix}$$

JPEG 2000

It is based on the same scheme as JPEG standard but unlike JPEG it is based on wavelet transform which is applied to the whole image (more precisely to its components). Wavelet transform is followed by quantization and lossless coding (entropy coding).

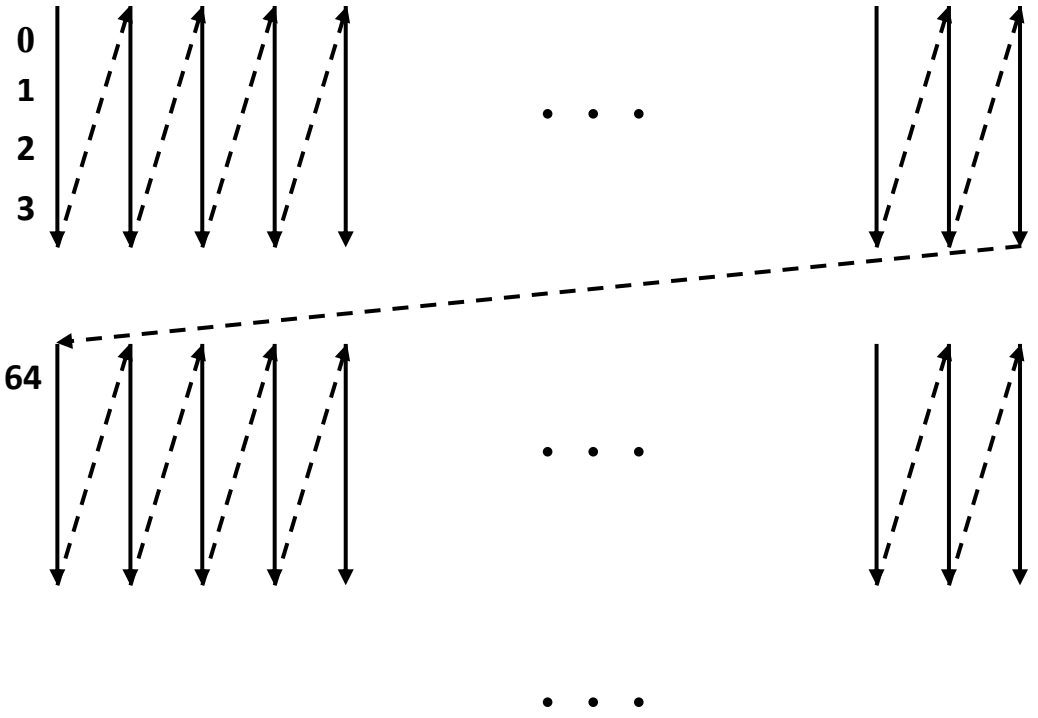
The important feature of JPEG 2000 is its coding scheme called Embedded Block Coding with Optimized Truncation (EBCOT).

Wavelet subband coefficients are partitioned into small blocks which in turn are coded by bit-plane, i.e. coefficient bits of the same order are coded together. The most significant bits are coded first, then low order bits are coded in descending order.

Binary context arithmetic coder is used to code bit-planes.

Context is formed using not only previously coded bits in the same bit-plane but also neighboring bits in the higher order bit-planes.

Bit-plane scanning order



JPEG2000

The passes are:

•Significance propagation (encode insignificant coefficients with at least one significant neighbor) (9 contexts (significance map)+ 5 contexts(sign)

•Magnitude refinement (encode significant coefficients) (3 contexts)

•Cleanup pass (encode insignificant coefficients with insignificant neighbors)

All passes use different contexts based on significance values of their 8 neighboring pixels. Cleanup uses additionally run-length context (all four coefficients in the column are insignificant and each has insignificant neighbors).

JPEG2000 provides spatial and SNR scalabilities.

Wavelet transform