Appendix 2A—Channel Bandwidth

Table 2A.1 summarizes the bandwidth of some important information storage and transmission channels encountered throughout this book.

Table 2A.1 Channel bandwidth.

Information Channel	Bandwidth	Comments
Standard telephone line	64 Kbps	
ISDN telephone line	P * 64 Kbps	P = 1 - 30
CD-ROM	1.248 Mbps	2 - 8X enhanced products
CD audio	1.4112 Mbps	
T-1 communications line	1.566 Mbps	
Digital video disc (DVD)	3.5 - 11 Mbps	Average - maximum rates
Ethernet LAN	10 Mbps	
Token-ring LAN	16 Mbps	
6-MHz terrestrial TV channel	20 Mbps	
DBS satellite transponder	27 Mbps	Transmits 4 - 8 channels
6-MHz cable TV channel	40 Mbps	
T-3 communications line	45 Mbps	
FDDI LAN	100 Mbps	
Fast ethernet LAN	100 Mbps	
ATM	155 Mbps	

mpeg 1 für CDROM-gemacht (Filme auf CD)

Appendix 2B—Media Characteristics

Tables 2B.1, 2B.2, and 2B.3 summarize the characteristics of various media encountered throughout this book.

Table 2B.1 Speech and audio.

Media	Resolution	Data Rate	Channel Bandwidth[1]	Compression Ratio[2]	Compression Algorithm
Telephone speech	8 KHz x 8 bits/sample	0.065 Mbps	< 0.065 Mbps	2 - 16:1	See Table 5.2(a)
Wideband speech	16 KHz x 8 bits/sample	0.131 Mbps	< 0.065 Mbps	2 - 4:1	See Table 5.2(b)
CD audio	2 x 44.1 KHz x 16 bits/sample	1.4112 Mbps	1.4112 Mbps	-	Uncompressed
CD MiniDisc™	2 x 44.1 KHz x 16 bits/sample	1.4112 Mbps	0.292 Mbps	5:1	ATRAC™, See Table 5.3
DCC™ audio tape	2 x 44.1 KHz x 16 bits/sample	1.4112 Mbps	0.384 Mbps[3]	4:1	PASC™, See Table 5.3
HDTV audio	5.1 (6) x 48 KHz x 16 bits/sample	4.608 Mbps	0.384 Mbps	12:1	MPEG-Audio, See Table 5.3

Notes:
[1] Bandwidth of existing channels and networks.
[2] Compression ratio objective to meet the limitations of existing channels and networks.
[3] Audio + ECC data = 0.768 Mbps.

Table 2B.2 Still image.

Media	Resolution[1]	Original Image	Typical Compressed Image	Typical Compression Ratio	Compression Algorithm
FAX	8.5 in. x 11 in. @ 200 x 200; 1 bit/pixel	3.74 Mb	0.250 Mb	15:1 verlustfrei	ITU-T T.4/6
Computer VGA (black-and-white)	640 x 480; 8 bits/pixel	2.46 Mb	0.123 Mb	20:1	JPEG and others
Computer SVGA (color)	1024 x 768; 16 bits/pixel	12.58 Mb	0.629 Mb	20:1	JPEG and others
Kodak Photo CD™ 35-mm slide	3072 x 2048; 24 bits/pixel	151 Mb	48 Mb[2]	4:1	Proprietary; See Section 9.9

Notes:
[1] Pixels/line x lines/image.
[2] The original image and four smaller images are stored at various resolutions and compression ratios. Each 650-MB CD-ROM contains 100 - 110 images.

Table 2B.3 Video.

Media	Resolution[1]	Data Rate	Channel Bandwidth[2]	Compression Ratio[3]	Compression Algorithm
Videophone (AT&T VideoPhone 2500™)	128 x 112; 12 bits/pixel @ 2-10 fps	0.344 - 1.72 Mbps	0.0192 Mbps (Analog telephone line)	18:1 - 90:1	Proprietary
Video-conferencing (ISDN videophone)	QCIF 176 x 144; 12 bits/pixel @ 10-30 fps	3.04 - 9.12 Mbps	128 Kbps (ISDN; P = 2)	23:1 - 70:1	H.261
Video-conferencing (high quality)	CIF 352 x 288; 12 bits/pixel @ 10-30 fps	12.2 - 36.5 Mbps	384 Kbps (ISDN; P = 6)	32:1 - 95:1	H.261
VCR-Quality (VHS equivalent MPEG-1 video)	SIF 352 x 240; 12 bits/pixel @ 30 fps	30.4 Mbps	1.248 Mbps (CD)[4]	25:1	MPEG-1
SDTV	MAIN profile 720 x 480 16 bits/pixel @ 30 fps	166 Mbps	4 Mbps (DBS channel)	42:1	MPEG-2
HDTV	HIGH profile 1920 x 1080 16 bits/pixel @ 30 fps	995 Mbps	15-20 Mbps (6-MHz TV channel)	50:1 - 66:1	MPEG-2

Notes:
[1] Pixels/line x lines/frame.
[2] Bandwidth of existing channels and networks.
[3] Compression ratio objective to meet the limitations of existing channels and networks.
[4] 1.248 Mbps of 1.4112 Mbps bandwidth is available for video.

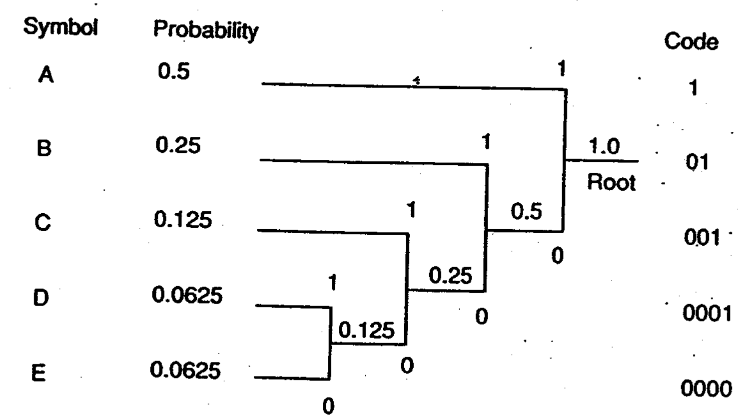
	Original source		Reduced source Stage 1		Reduced source Stage 2		Reduced source Stage 3
s_i	$p(s_i)$	s'_i	$p(s'_i)$	s''_i	$p(s''_i)$	s'''_i	$p(s'''_i)$
s_1	0.40	s'_1	0.40	s''_1	0.40	s'''_1	0.60
s_2	0.20	s'_2	0.25	s''_2	0.35	s'''_2	0.40
s_3	0.15	s'_3	0.20	s''_3	0.25		
s_4	0.15	s'_4	0.15				
s_5	0.10						

a) Source reduction process

	Original source		Reduced source Stage 1		Reduced source Stage 2		Reduced source Stage 3
s_i	Codeword	s'_i	Codeword	s''_i	Codeword	s'''_i	Codeword
s_1	1	s'_1	1	s''_1	1	s'''_1	0
s_2	000	s'_2	01	s''_2	00	s'''_2	1
s_3	001	s'_3	000	s''_3	01		
s_4	010	s'_4	001				
s_5	011						

b) Codeword construction process

Figure 3.1: Huffman code generation.



(a) Code tree construction

Symbols	AECBD
Huffman code	10000001010001
Parsed	1 | 0000 | 001 | 01 | 0001

(b) Encoding and decoding

Figure 4.3 Huffman coding example.

$$y = 0 \tag{2.24a}$$
$$y = a \tag{2.24b}$$
$$y = b \tag{2.24c}$$
$$y = c \tag{2.24d}$$
$$y = a + b - c \tag{2.24e}$$
$$y = a + \frac{b-c}{2} \tag{2.24f}$$
$$y = b + \frac{a-c}{2} \tag{2.24g}$$
$$y = \frac{a+b}{2} \tag{2.24h}$$

Note that, pixel values at pixel positions a, b, and c are available to both the

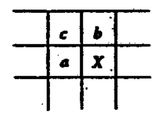
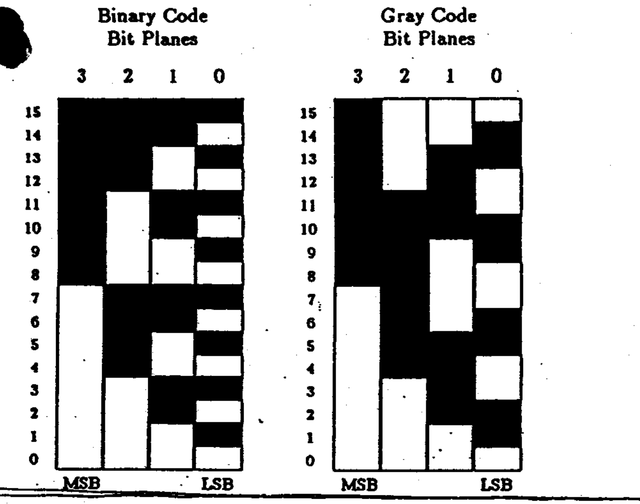


Figure 2.9 Lossless JPEG prediction kernel.

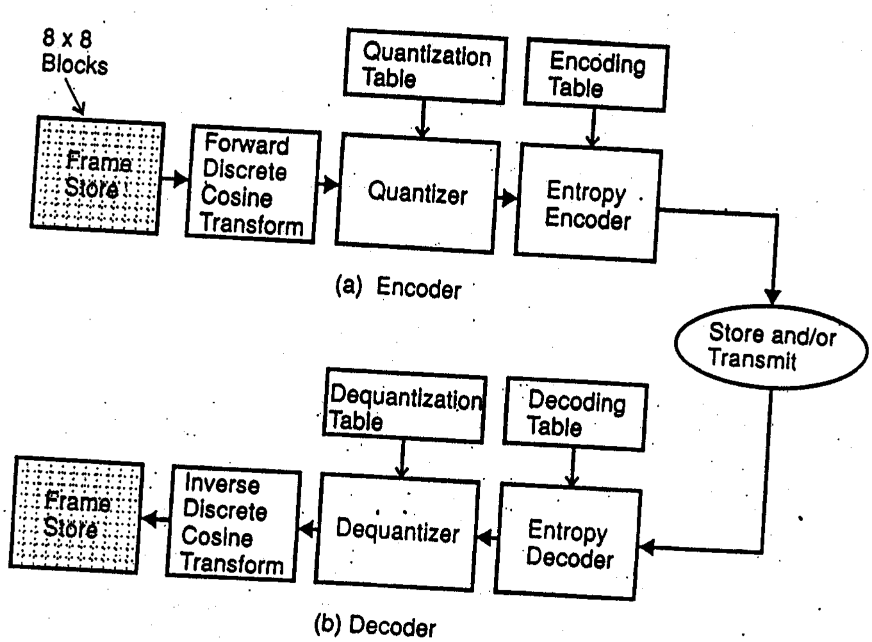
Category	Prediction Residual
0	0
1	-1, 1
2	-3, -2, 2, 3
3	-7, ..., -4, 4, ..., 7
4	-15, ..., -8, 8, ..., 15
5	-31, ..., -16, 16, ..., 31
6	-63, ..., -32, 32, ..., 63
7	-127, ..., -64, 64, ..., 127
8	-255, ..., -128, 128, ..., 255
9	-511, ..., -256, 256, ..., 511
10	-1023, ..., -512, 512, ..., 1023
11	-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

Table 2.9 Prediction residual categories for lossless JPEG compression.



...re 6.2: Binary and Gray code bit planes for LENA, (BP 7 through 4).

The Existing DCT-Based JPEG Standard



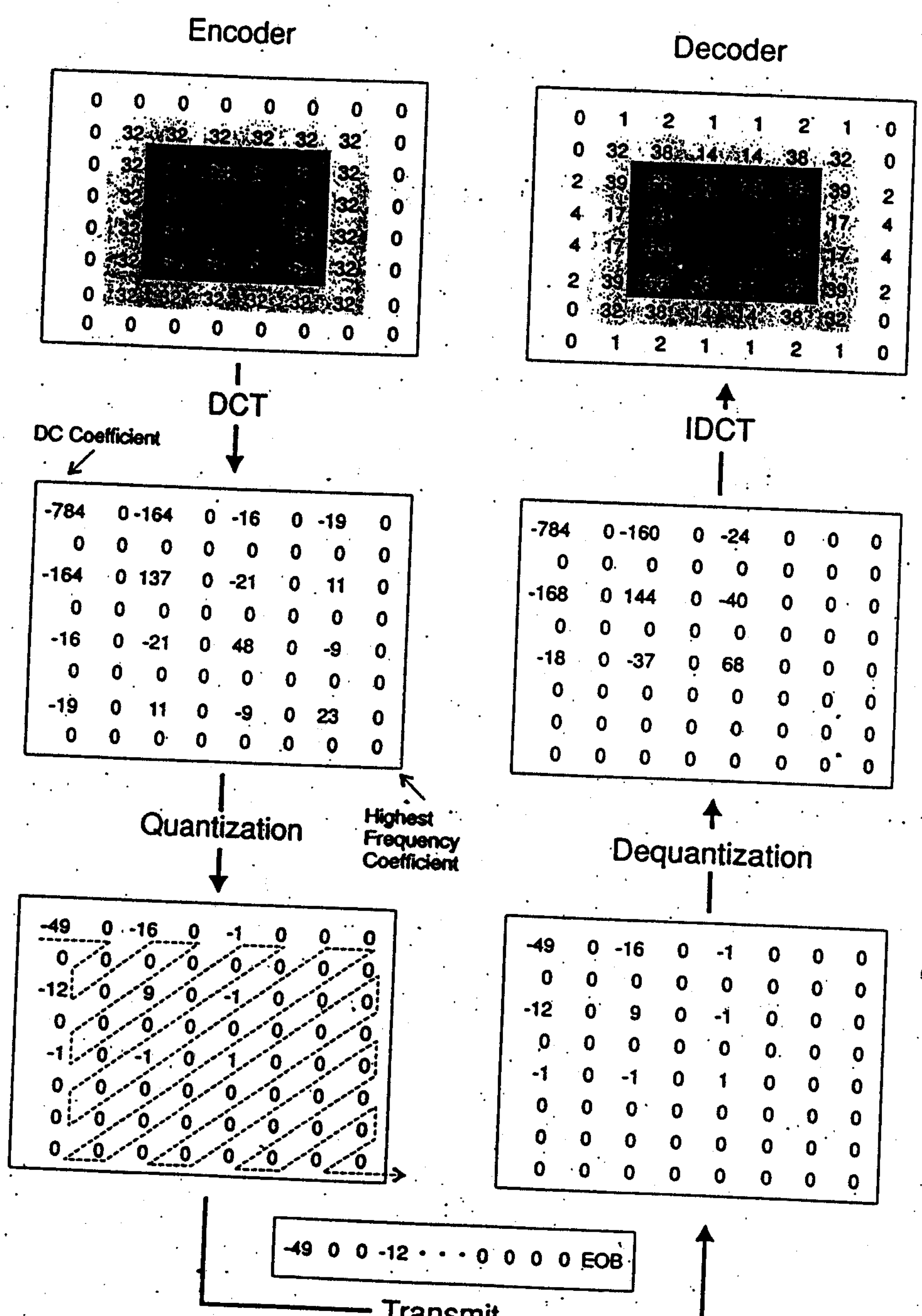
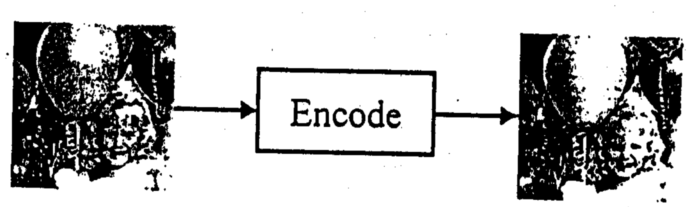


Figure 5.10 JPEG 8 × 8 pixel block coding.

The Existing JPEG Standard "Toolkit"

- The existing JPEG standard concerns with the compression of continuous-tone, still-frame, monochrome and color images. It provides a "toolkit" of compression techniques from which applications can select the elements that satisfy their particular requirements.
  - **Baseline system**: A simple and efficient DCT-based algorithm that uses Huffman coding, operates only in sequential mode, and is restricted to 8 bits/pixel input.
  - **Extended system**: Enhancements to the baseline to satisfy broader applications.
  - **Lossless mode**: Based on predictive coding and independent of the DCT that uses either Huffman or arithmetic coding.

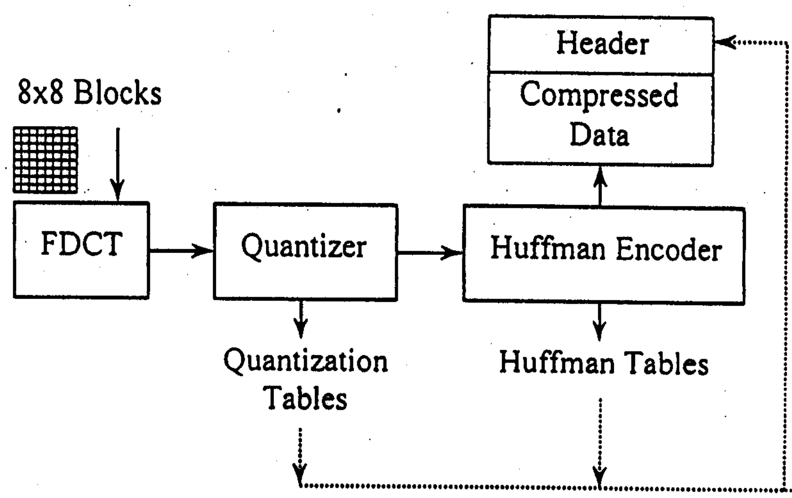
Old Compression Paradigm (JPEG Baseline)



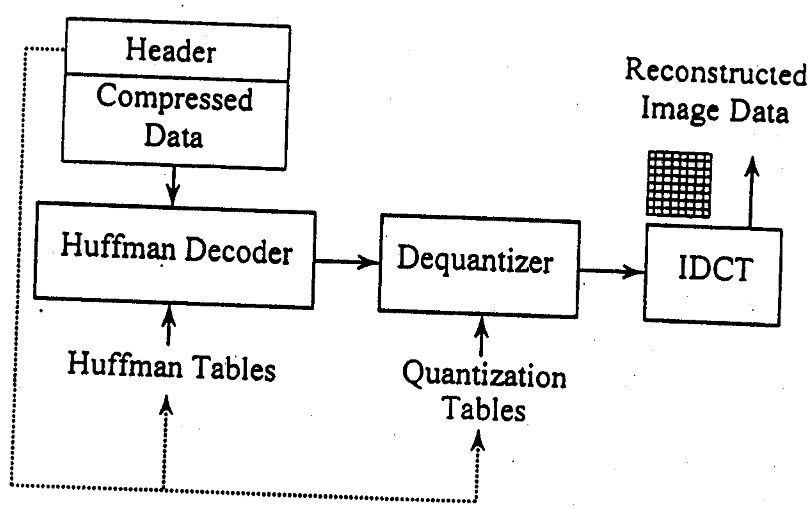
<u>Encoder choices</u>
color space
quantization
entropy coder
pre-processing

<u>No decoder choices</u>
only one image
post-processing

JPEG Encoder Block Diagram



JPEG Decoder Block Diagram



JPEG Encoding Example: Original Block

The image is first segmented into 8 x 8 blocks. Each block is encoded independent of the other blocks (except for the DC coefficient of the DCT transform). Following is an example 8 x 8 block from the Lena image.

136	142	151	165	180	196	201	210
134	142	152	165	174	195	202	209
131	139	156	163	176	190	200	210
132	141	150	156	172	189	197	214
133	139	146	158	168	187	200	214
133	137	144	157	167	186	202	211
127	136	144	159	166	187	203	208
130	139	146	158	165	185	197	210

Level-Shifted Block

The value of 128 is subtracted from each pixel prior to the application of the discrete cosine transform (DCT). This places the **DC coefficient** (the top-left corner coefficient that is 8 times the average brightness of the block) in the range (-1024, +1016).

$f(j,k) =$

8	14	23	37	52	68	73	82
6	14	24	37	46	67	74	81
3	11	28	35	48	62	72	82
4	13	22	28	44	61	69	86
5	11	18	30	40	59	72	86
5	9	16	29	39	58	74	83
-1	8	16	31	38	59	75	80
2	11	18	30	37	57	69	82

Discrete Cosine Transform (DCT)

- The heart of both the JPEG and the MPEG family of standards is the DCT operation.
- For each 8 x 8 block of an 8-bit input image, the forward DCT produces an 8 x 8 set of 11-bit coefficients in the range (-1024, 1024).

$$F(u,v)=\frac{C(u)C(v)}{4}\sum_{j=0}^{7}\sum_{k=0}^{7} f(j,k)\cos\left[\frac{(2j+1)u\pi}{16}\right]\cos\left[\frac{(2k+1)v\pi}{16}\right]$$

$$\text{where}\quad c(w)=\begin{cases}\frac{1}{\sqrt{2}} & \text{if}\quad w=0\\ 1 & \text{otherwise}\end{cases}$$

DCT Of 8x8 Image Block

The DCT coefficients are much less correlated, and due to its energy compaction property, this operation redistributes the signal energy among only a few coefficients.

DC Value

$F(u,v)=$

327.5	-215.8	16.1	-10.7	-3.7	-1.5	4.2	-6.7
18.1	3.4	-9.9	3.7	0.5	-3.2	3.5	2.2
2.5	1.3	-5.4	2.8	-1.0	2.3	-1.6	-2.6
0.6	-2.5	3.0	5.0	1.8	2.2	-2.6	-1.4
0.3	1.6	3.4	0.0	2.5	-5.1	1.6	-0.7
-0.6	-1.8	-2.4	0.5	-0.4	-1.6	-0.1	2.1
0.9	1.6	-0.6	-0.7	2.1	-0.5	0.9	2.8
0.6	-1.0	-2.9	-1.4	0.2	1.9	-0.6	0.7

DCT Coefficient Quantization

- Each DCT coefficient is uniformly quantized with a quantization step that is taken from a user-defined **quantization table** (q-table or normalization matrix), characterized by 64, 1-byte elements.
- The quality and compression ratio of an encoded image can be varied by changing the q-table elements (usually by scaling up or down the values of an initial q-table).
- The q-table is often designed according to the perceptual importance of the DCT coefficients (e.g., by using the HVS CSF data) under the intended viewing conditions.
- For the baseline system, in order to meet the needs of the various color components, four different quantization tables are allowed.

Quantization Matrix

The JPEG committee has used the following quantization matrix (based on psychophysical studies) to quantize luminance data as an example in their draft proposal.

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

The Independent JPEG Group (IJG)

- The IJG code controls the image quality (or compression ratio) by scaling a reference q-table with a user-selectable factor Q, referred to as the "**quality factor**".
- The value of Q spans the range of 0 (lowest quality) to 100 (highest quality). At Q = 50, the reference table is scaled by 1. For values of Q in the range of (0-100):
  - If Q < 50, then Q = 5000/Q
  - Otherwise, Q = 200 - 2*Q
- The value of Q expressed in percent is used to scale the q-table, e.g., if Q = 20, the q-table elements are multiplied by 1.60. For Q = 100, all the q-table elements are set to 1.

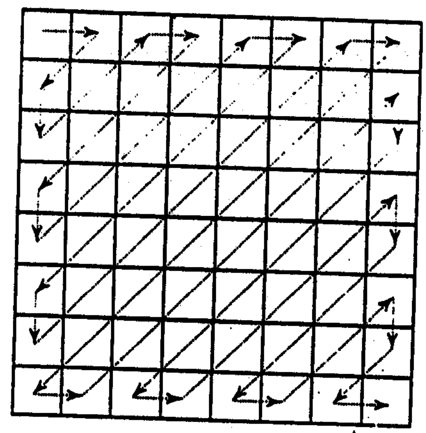
Quantized DCT Coefficients

For typical blocks in an image, the process of normalization followed by quantization results in many zero-valued coefficients that can be coded efficiently.

20	-20	2	-1	0	0	0	0
2	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	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

ZigZag Pattern

After quantization, the DCT coefficients are reordered into a 1-D format using a zig-zag pattern. This rearranges the coefficients approximately in the order of decreasing energy.



Coding Of Quantized Coefficients

- 20 -20 2 0 0 2 -1 -1 EOB
- The zig-zag reordering creates long runs of zeros that can be efficiently encoded by an end-of-block (EOB) symbol.
- The difference between the quantized DC coefficient of the current block and the one from the previous block is Huffman coded.
- The AC coefficients are encoded using Huffman coding on magnitude/runlength pairs, i.e., the magnitude of a nonzero AC coefficient plus the runlength of zero-valued AC coefficient preceding it is encoded by a Huffman code.

Inverse Discrete Cosine Transform

The resulting coefficients are then inverse transformed according to the following equation:

$$f(j,k) = \sum_{u=0}^{7} \sum_{v=0}^{7} C(u)C(v)F(u,v)\cos\left[\frac{(2j+1)u\pi}{16}\right]\cos\left[\frac{(2k+1)v\pi}{16}\right]$$

where $C(w) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{for } w = 0, \text{ and,} \\ 1, & \text{otherwise.} \end{cases}$

The value of 128 is added to the result of the inverse transform to reconstruct an approximation to the original block.

Dequantized DCT Coefficients

At the receiver, the normalized and quantized DCT coefficients are inverse scaled (dequantized) to the proper range by multiplying them by the corresponding quantization table values.

320	-220	20	-16	0	0	0	0
24	0	-14	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	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Reconstructed 8x8 Image Block

Due to the energy preserving nature of the DCT, the root-mean-squared-error (RMSE) between the original image block and the reconstructed block is the same in both the image domain and the DCT domain.

132	140	153	166	178	191	204	213
132	140	152	165	177	190	204	213
131	139	151	163	175	189	203	212
131	138	149	160	172	187	202	212
131	137	147	157	169	184	201	212
130	136	145	155	167	182	200	211
130	135	143	153	165	181	199	211
130	135	142	151	163	180	198	211

Difference (Error) Image

The RMSE between the original image block and the reconstructed block (standard deviation of the error) for this example is 2.84 codevalues. The signal-to-noise-ratio (SNR) is defined as 20 $\log_{10}(255/\text{RMSE})$ in db units.

4	2	-2	-1	2	5	-3	-3
2	2	0	0	-3	5	-2	-4
0	0	5	0	1	1	-3	-2
1	3	1	-4	0	2	-5	2
2	2	-1	1	-1	3	-1	2
3	1	-1	2	0	4	2	0
-3	1	1	6	1	6	4	-3
0	4	4	7	2	5	-1	-1

JPEG Baseline Summary

- Baseline JPEG is a lossy compression technique. The image quality can be traded for lower bit rate by manipulating the normalization matrix components.
- Using the same compression parameters with different images would generally result in roughly the same image quality but in different compression ratios.
- The encoder and decoder have roughly the same complexity.
- At low bit rates, the DCT results in blocking artifacts in addition to ringing around the edges.
- Due to Huffman coding, channel errors can potentially have a catastrophic effect.

JPEG Extended System

- 12-bit/pixel input image precision
- Sequential progressive build-up
- Hierarchical progressive build-up
- Arithmetic coding (instead of Huffman)
- Adaptive quantization (similar to MPEG)
- Simple and composite tiling of large images
- Selective refinement

Grundlegendes

- Fourier Transformation:

$$W_\omega(f) = \int_{-\infty}^{\infty} f(t)e^{-it\omega}dt$$

- Nur globale Frequenzinformation ! Keine lokale Information möglich !
- Abhilfe: gefensterte Fouriertransformation. Probleme: konstante Fensterbreite, Heisenberg'sche Unschärferelation
- Wavelet Transformation:

$$W_{a,b}(f) = |a|^{-1/2} \int_{-\infty}^{\infty} f(t)\, \psi\left(\frac{t-b}{a}\right)\, dt$$

- 2 Parameter: Translation (b) und Dilatation (a)
- ψ heißt "Motherwavelet"
- Diskretisierung: $a = a_0^m$, $b = nb_0a_0^m$, z.B. $a_0 = 2$ and $b_0 = 1$
- Diskrete Wavelettransformation:

$$W_{m,n}(f) = 2^{-m/2} \int_{-\infty}^{\infty} f(t)\psi(2^{-m}t - n)\, dt$$

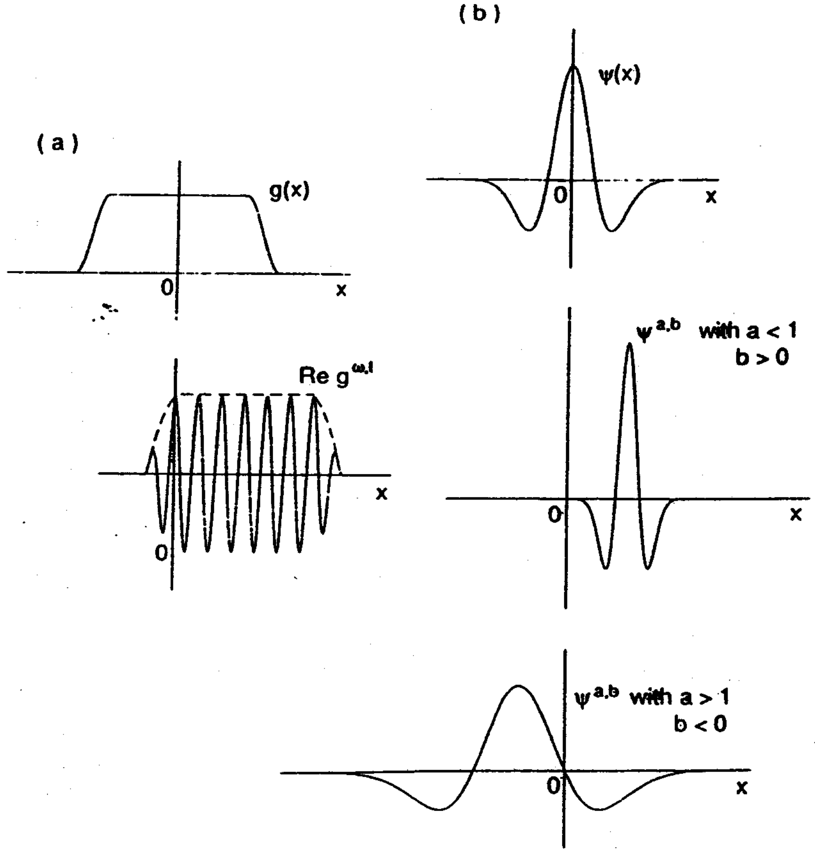


FIG. 1.2. *Typical shapes of* (a) *windowed Fourier transform functions* $g^{\omega,t}$, *ana* b) *wavelets* $\psi^{a,b}$. *The* $g^{\omega,t}(x) = e^{-i\omega x}g(x-t)$ *can be viewed as translated envelopes* g *"filled in" with higher frequencies; the* $\psi^{a,b}$ *are all copies of the same functions, translate and compressed or stretched.*

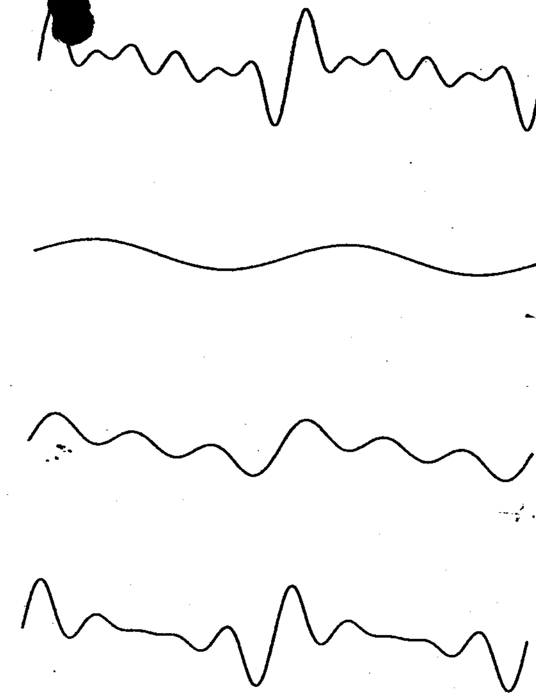


FIG. 1.1. *Original signal and its two-, four- and six-term Fourier expansions.*

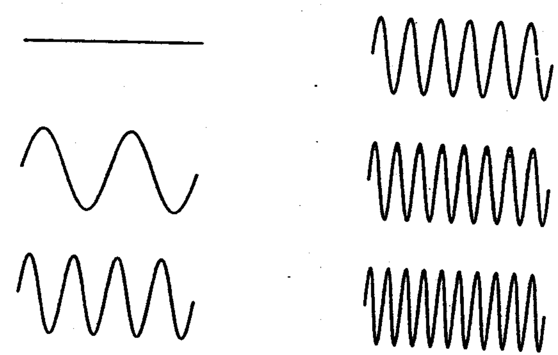


FIG. 1.2. *Trigonometric basis functions.*

FIG. 1.3. *Original signal.*

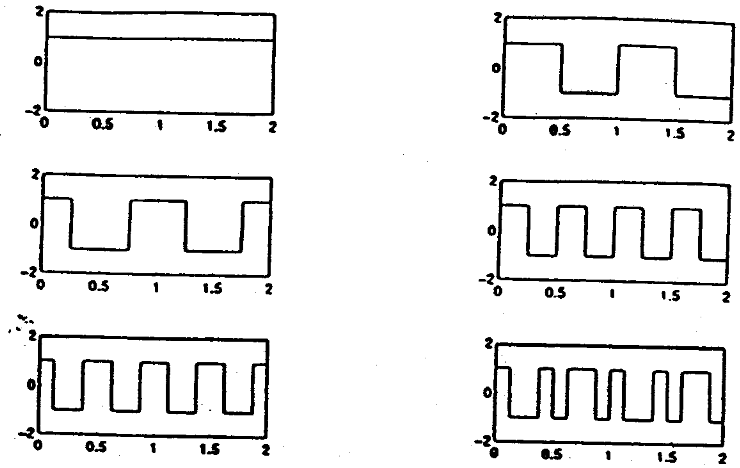


FIG. 1.4. *Walsh basis functions.*

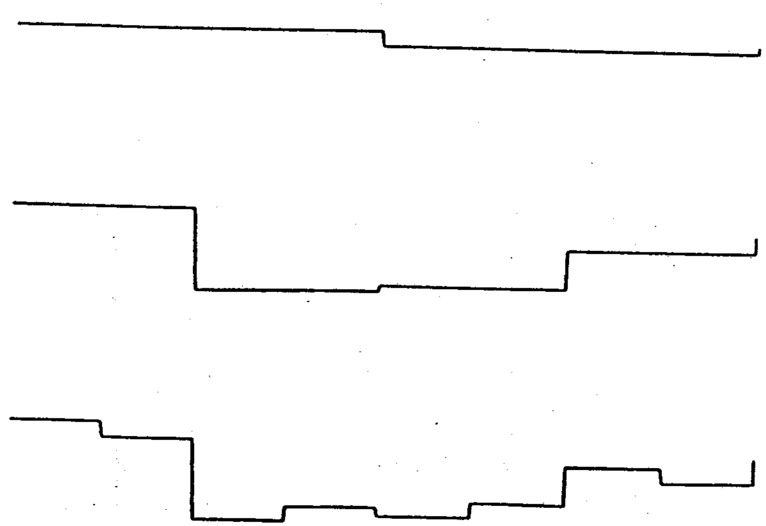


FIG. 1.5. *Model of original signal using the two-, four- and six-term Walsh series.*

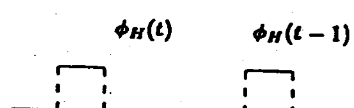


FIG. 1.6. *Haar scaling functions.*

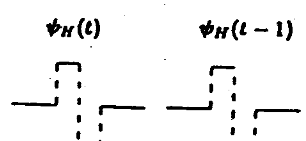


FIG. 1.7. *Haar wavelets.*

Observe that the mathematical formulation of the Fourier transform alone does not give rise to any localization capability. In Figure 1.12, we show the magnitude of the Fourier transform of the music data displayed in Figure 1.11. This musical piece contains not only very high frequency values, but, unfortunately, certain high-frequency noise is also mixed in with the music and occurs in the same frequency ranges. If a standard lowpass filter is applied to remove the noise, the high-frequency music content is removed as well. This is shown in Figures 1.13 and 1.14. The smooth curve in Figure 1.13 means that the high-frequency content of the music has been removed, and the music then sounds flat. For comparison, we plot the IWT of the same noisy music data in Figure 1.15, where the frequency axis of the time-frequency plane is matched with the frequency domain that displays the Fourier transform of the data. Observe that since the noise content has larger amplitude than the music itself on the same frequency ranges, it shows up much more prominently. In Figure 1.16, we also show the DWT of the same music data. The rectangular structure indicates that only discrete values of the IWT are obtained, but they

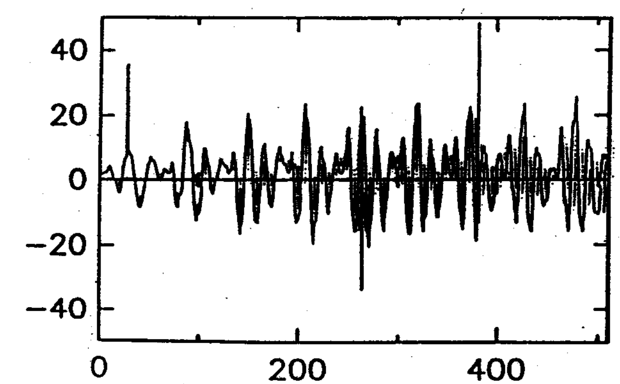


FIG. 1.11. *Music data with high-frequency (pop) noise.*

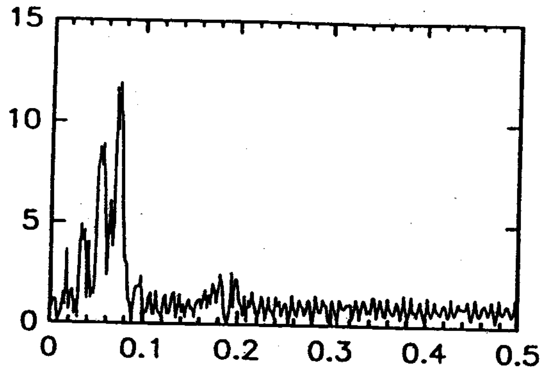


FIG. 1.12. *Magnitude of the Fourier transform of music data.*

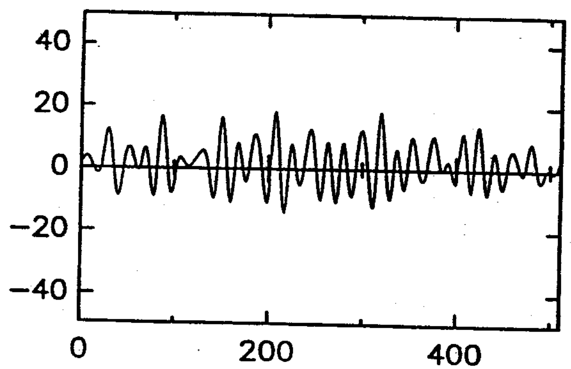


FIG. 1.13. *Music data with high-frequency noise (and music) removed.*

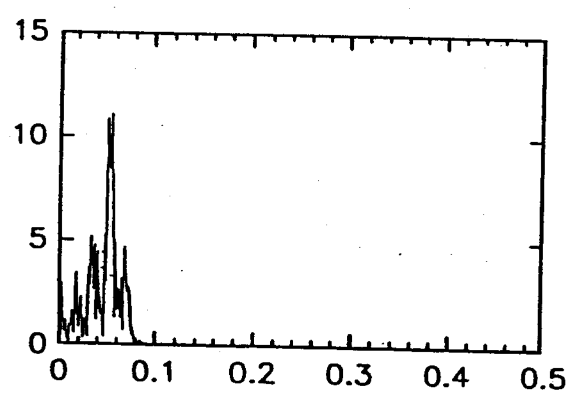


FIG. 1.14.

In Figure 1.17, we demonstrate the concept of wavelet decomposition of (a longer piece of) the same music data modeled by $f_n(t)$. The signal $f_n(t)$ shown at the top left-hand corner is decomposed as the sum of the $f_{n-1}(t)$ and $g_{n-1}(t)$ components, shown in the second row. The third row is the decomposition of $f_{n-1}(t)$ as the sum of the $f_{n-2}(t)$ and $g_{n-2}(t)$, etc. Observe that the large amplitudes of the noise content in $g_{n-1}(t)$ and $g_{n-2}(t)$ are quite prominent. After truncating these values, we can reconstruct the signal as shown in Figure 1.18. Here, while the high-frequency music is unchanged, only the high-frequency noise has been removed.

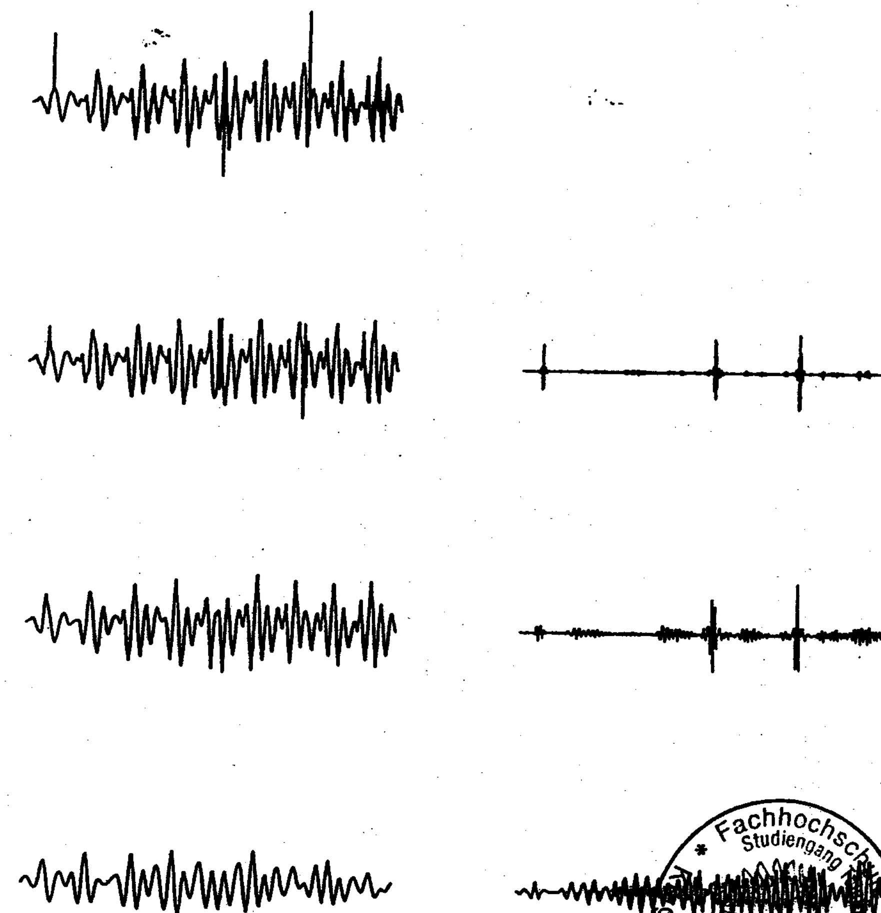


FIG. 1.17. *Wavelet decomposition of music data.*

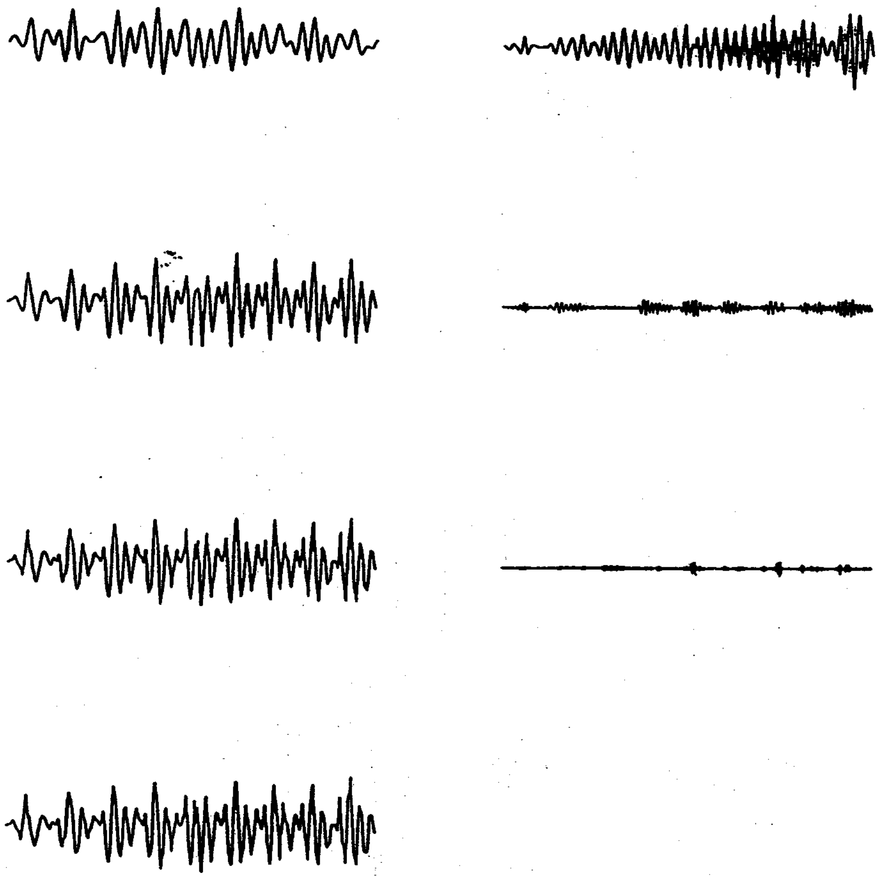


FIG. 1.18. *Wavelet reconstruction of music data after removal of pop noise.*

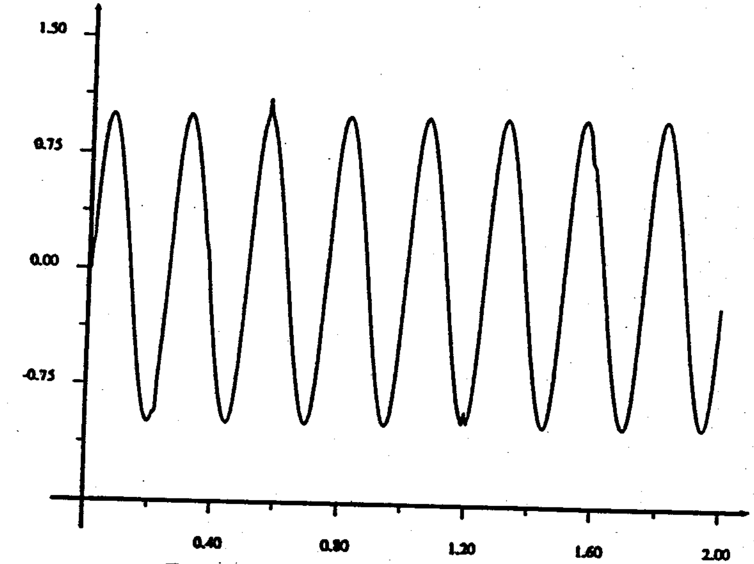


FIG. 1.19. *Perturbed sinusoidal curve.*

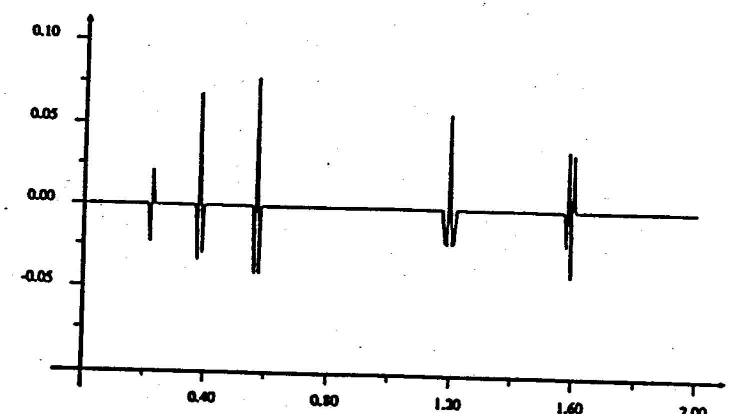


FIG. 1.20. *Perturbation magnified after sinusoidal curve is removed.*

As another example, a sinusoidal curve has been slightly perturbed and is represented by $f_n(t)$ in Figure 1.19. This perturbation shows up very clearly as $g_{n-1}(t)$ in Figure 1.20. The low-frequency content (of the pure sinusoidal signal) occurs in the DC component $f_{n-1}(t)$ but is not shown here. This example indicates the importance of the DWT in signal detection. In the next section, we will show that the computation of the DWT is even faster than that of the fast Fourier transform (FFT).

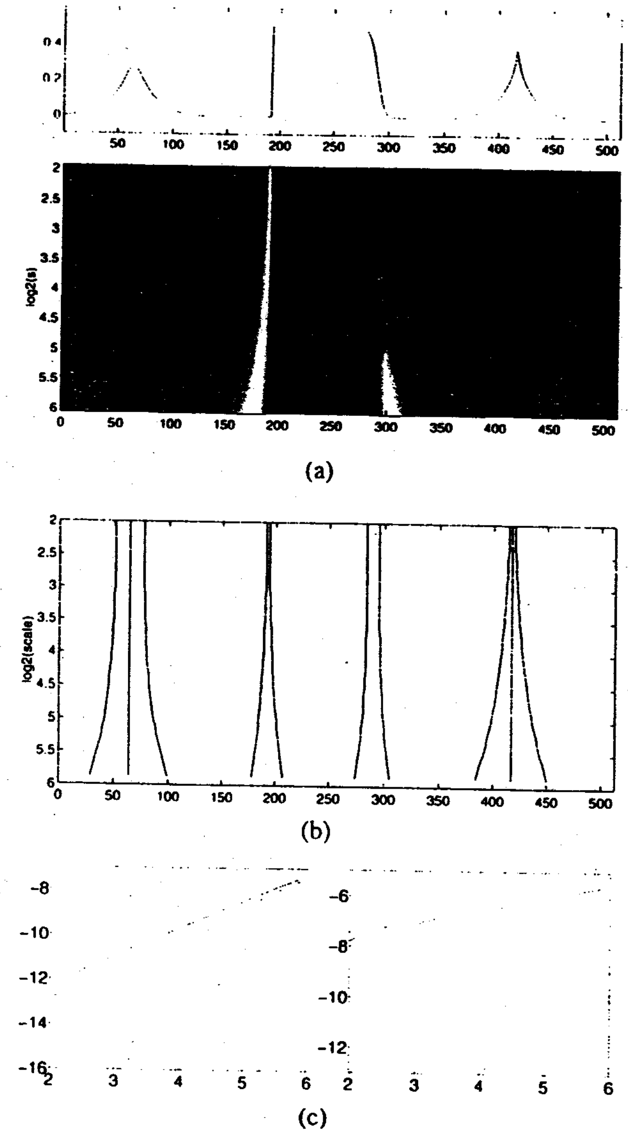


FIGURE 6.6 (a): Wavelet transform $Wf(u,s)$ as a function of u and $\log_2 s$. (b): Modulus maxima of a wavelet transform computed $\psi = \theta''$, where θ is a Gaussian with variance $\beta = 1$. (c): Decay of $\log_2 |Wf(u,s)|$ as a function of $\log_2 s$ along maxima curves. In the left figure, the solid and dotted lines correspond respectively to the maxima curves converging to $t = 416$ and $t = 64$. In the right figure, they correspond respectively to the curves converging to $t = 193$ and $t = 280$. The diffusion at $t = 64$ and $t = 280$ modifies the decay for $s \leq \sigma = 2^4$.

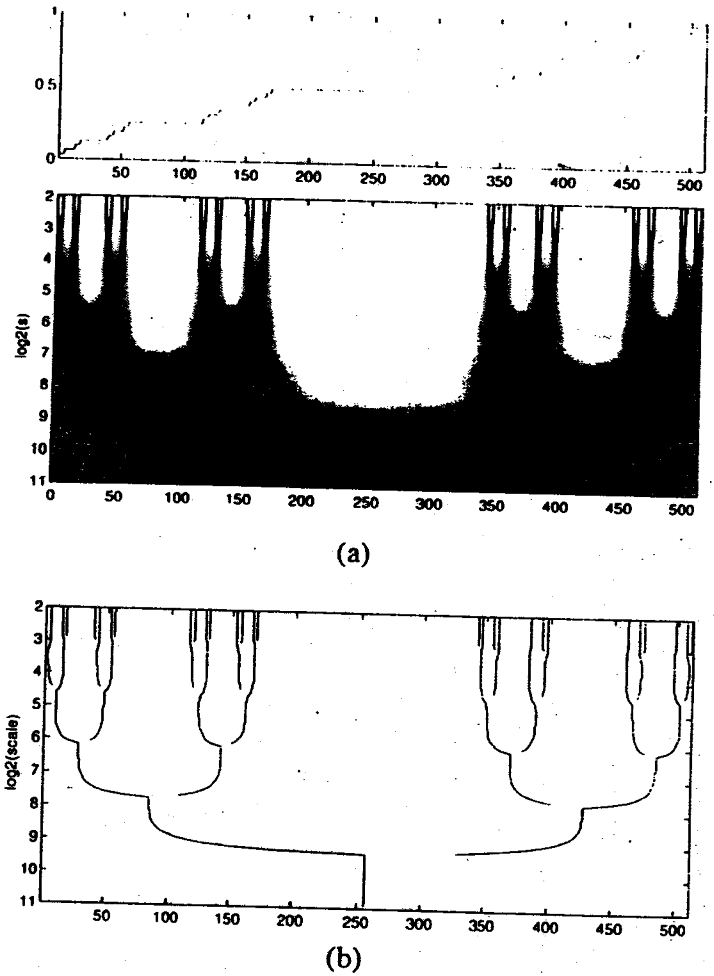


FIGURE 6.16 Devil's staircase calculated from a Cantor measure with equal weights $p_1 = p_2 = 0.5$. (a): Wavelet transform computed with $\psi = -\theta'$, where θ is Gaussian. (b): Wavelet transform modulus maxima.

$$f(t) = \begin{cases} p_1\, f(3t) & \text{if } t \in [0, \frac{1}{3}] \\ p_1 & \text{if } t \in [\frac{1}{3}, \frac{2}{3}] \\ p_1 + p_2\, f(3t-2) & \text{if } t \in [\frac{2}{3}, \ldots \end{cases}$$

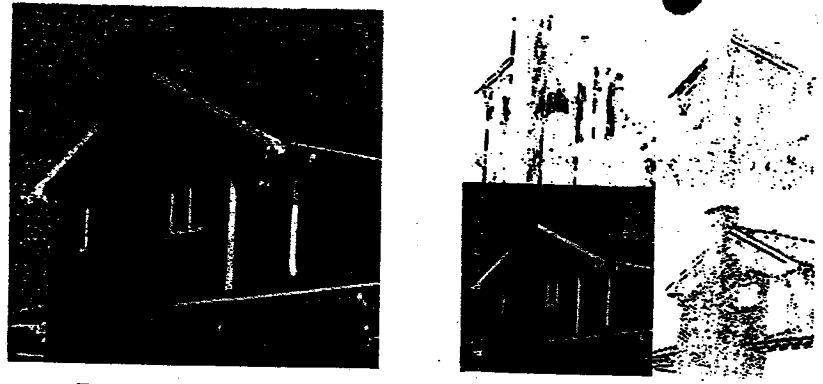


FIG. 7.3. *The house image and its first-level wavelet decomposition.*

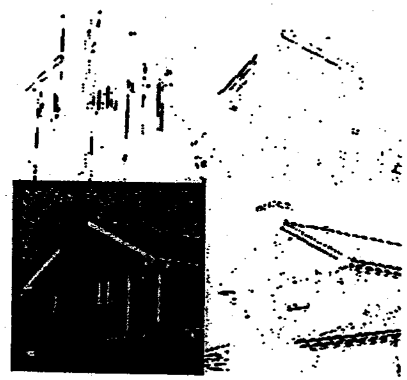


FIG. 7.4. *Thresholded and magnified details of the first-level DWT.*

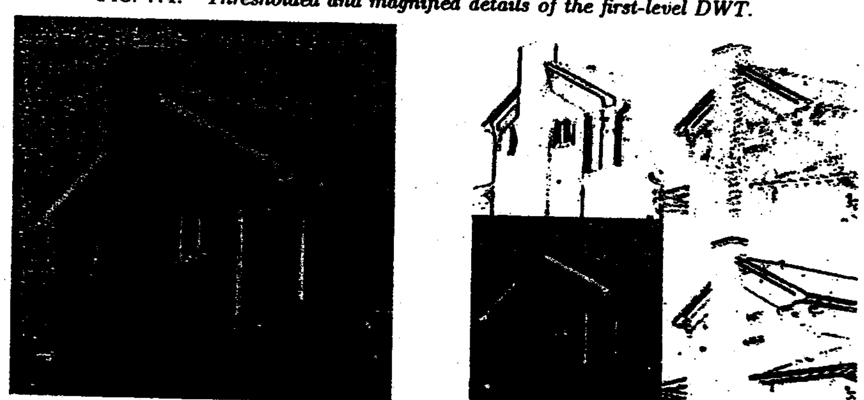


FIG. 7.5. *The LL subimage is magnified four times using pixel replication and is treated as an original image, and the DWT of this subimage is shown on the right.*

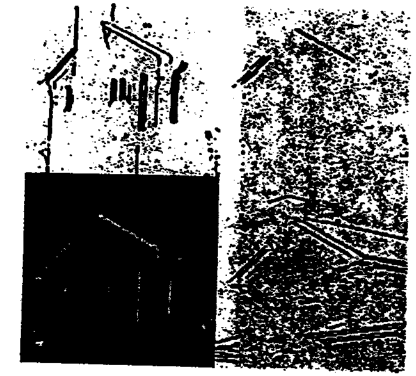


FIG. 7.6. *Thresholded and magnified details of the second-level DWT.*

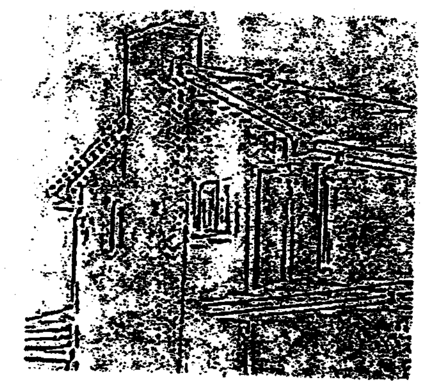


FIG. 7.7. *Superposition of the details from Figures 7.4 and 7.6.*

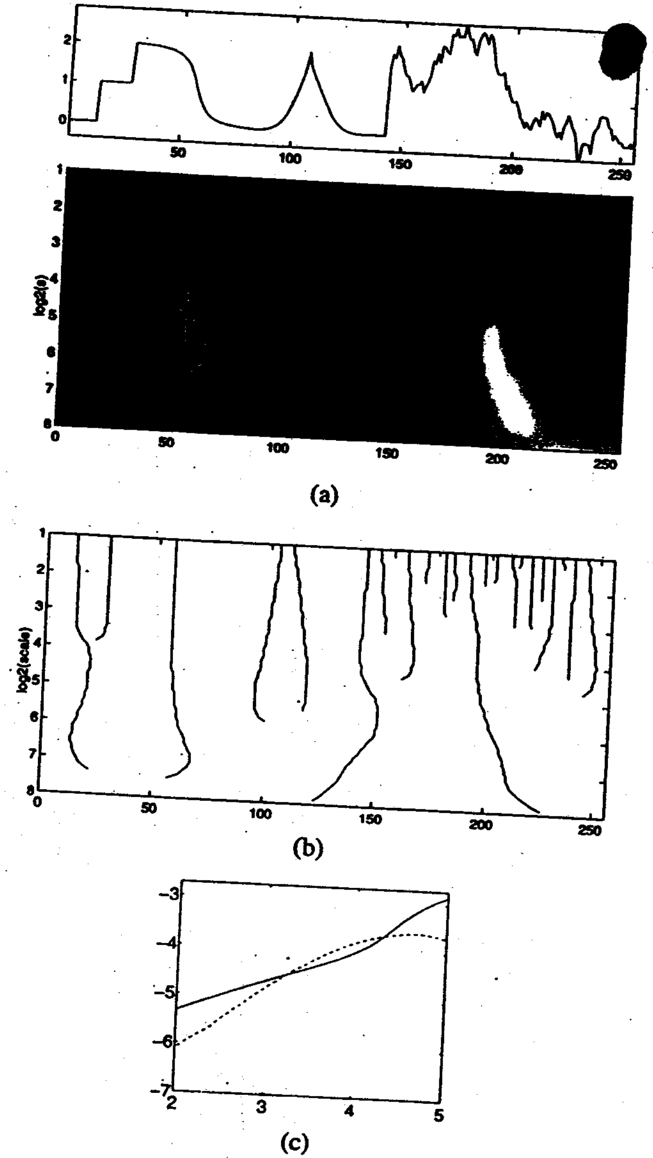
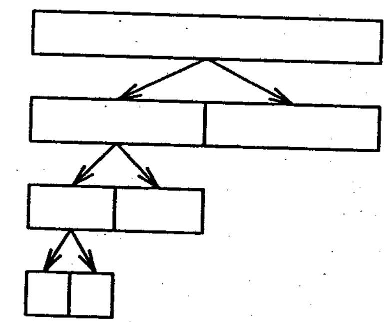
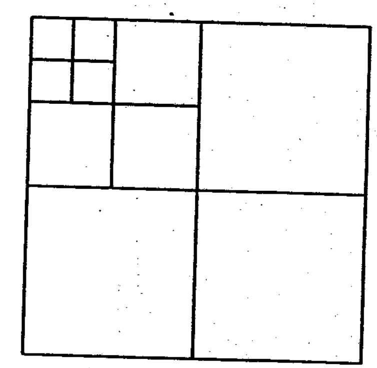


FIGURE 6.5 (a): Wavelet transform $Wf(u,s)$. The horizontal and vertical axes give respectively u and $\log_2 s$. (b): Modulus maxima of $Wf(u,s)$. (c): The full line gives the decay of $\log_2 |Wf(u,s)|$ as a function of $\log_2 s$ along the maxima line that converges to the abscissa $t = 14$. The dashed line gives $\log_2 |Wf(u,s)|$ along the left maxima line that converges to $t = 108$.

Schnelle Algorithmen

1. Multiresolution Analysis
2. Komplexität: O(N) Bem: FFT O(N logN) !!





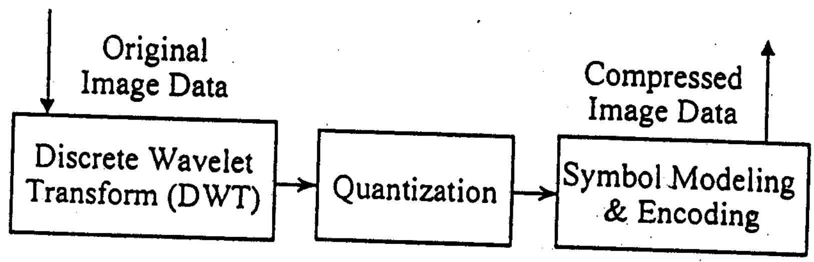
Wavelet Compression

Wavelet Image Compression

- The image is decomposed into a number of subbands, each representing a different spatial frequency, by recursively filtering with a low-pass/high-pass filter pair (known as **analysis filters**), followed by down sampling. The filters are chosen so that minimal or no information is lost in the decimation process.
- The subbands are independently or jointly quantized and coded using conventional techniques (e.g., scalar quantization, Huffman or arithmetic coding, etc.).
- At the decoder, the reconstructed subbands are upsampled and interpolated using a bank of low-pass and high-pass filters known as **synthesis filters** that are derived from the analysis filters so as to minimize (or completely eliminate) any artifacts resulting from the analysis and synthesis stages.

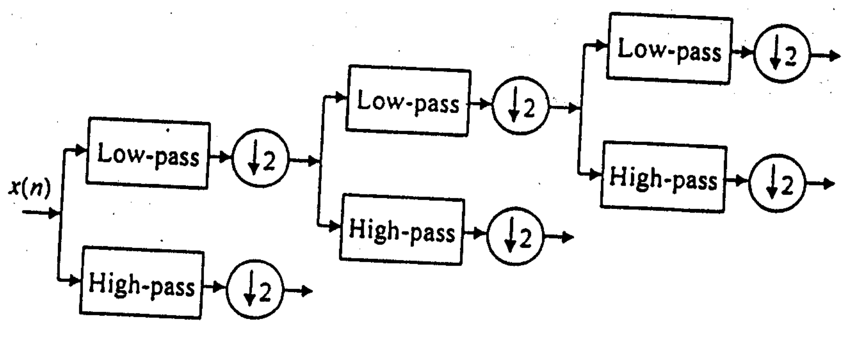
Wavelet Encoder Block Diagram

- A wavelet compression scheme uses a discrete wavelet transform (DWT) as the transformation/decomposition component of the compression system.



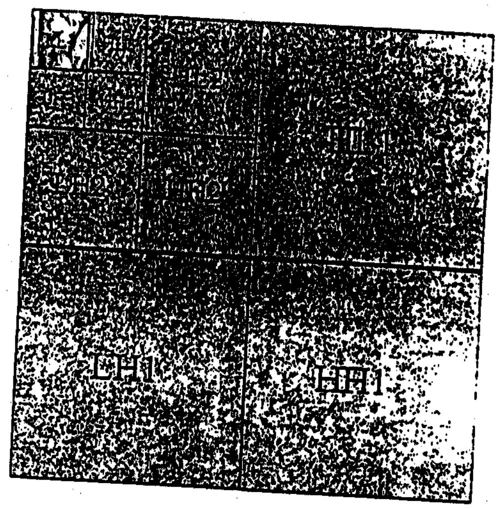
Analysis Filter Bank

Following is an example of a 3-level, 1-dimensional analysis filter bank. For 2-D images, 1-D filters are applied in succession to the rows and the columns of the image.



Example Of Wavelet Decomposition

A 3-level wavelet decomposition of Lena using the Daubechies (9,7) filter bank with a normalization of 1. The high-frequency subbands have been scaled up by a factor of 4, while the DC (or the LL) band has been scaled down by a factor of 2.



Bi-Orthogonal Filter Banks

- Most wavelet based image compression systems use a class of analysis/synthesis filters known as **bi-orthogonal** filters:
  - The basis functions for $h_0(n)$ and $g_1(n)$ are orthogonal; and the basis functions for $h_1(n)$ and $g_0(n)$ are orthogonal.
  - Linear-phase (symmetrical) and perfect reconstruction.
  - Unequal length; odd-length filters differ by an odd multiple of two (e.g., 7/9), while even-length filters differ by an even multiple of two (e.g., 6/10).
  - Symmetric boundary extension.
  - Possible regularity, vanishing moments property.

Bi-Orthogonal Filter Symmetry Conditions

- Let h_0 denote the low-pass finite impulse response (FIR) analysis filter, and h_1 denote the high-pass FIR analysis filter in a bi-orthogonal filter bank. Two situations occur:
  - Both filters are odd-length. Then h_0 and h_1 are both symmetric and are referred to as **whole-sample symmetric** (WSS) filters.
  - Both filters are even length. Then h_0 *is* symmetric and h_1 is anti-symmetric and are referred to, respectively, as **half-sample symmetric** (HSS) and **half-sample anti symmetric** (HSA).
- Due to symmetry conditions, only the transmission of half of the filter coefficients are necessary.

Example Of Bi-Orthogonal Filters

n=0

h_0	9	0.6029490, 0.2668641, -0.0782232, -0.0168641, 0.0267487
h_1	7	1.1150870, -0.5912717, -0.0575435, 0.0912717
h_0	6	0.5575435, 0.0337282, - 0.0912717
h_1	10	0.8698131, -0.1886408, -0.0950873, -0.0098846, 0.0267487
h_0	13	CRF (13, 7) (164, 80, -31, -16, 14, 0, -1) / 256
h_1	7	(1, -9, 0, 1) / 16
h_0	13	Swelden (13, 7) (348, 144, -63, -16, 18, 0, -1) / 512
h_1	7	(1, -9, 0, 1) / 16
h_0	2	1/2
h_1	10	(1, -22, -22, 3, 3) / 128

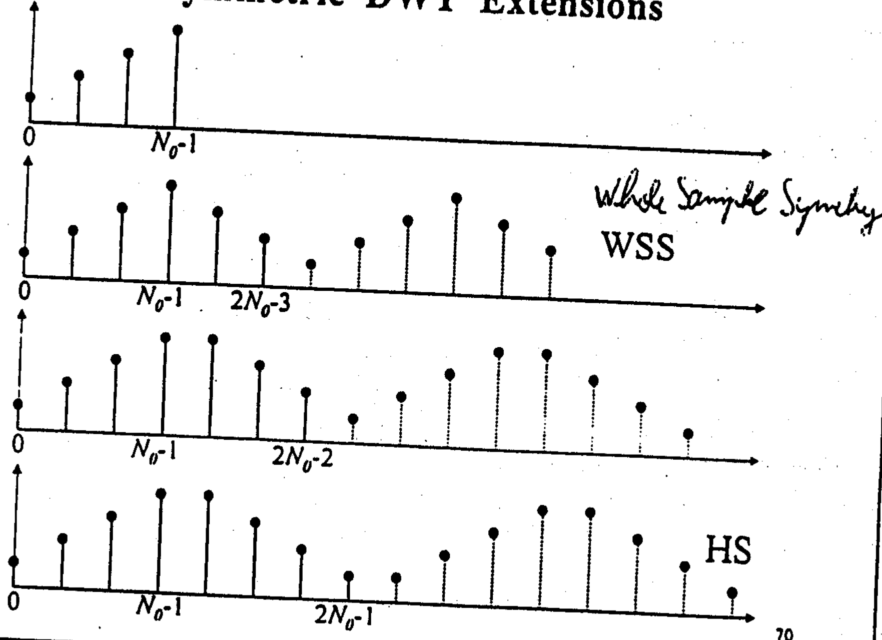
h_0	2	1/2
h_1	6	(8, -1, -1) / 8

h_0	5	(6, 2, -1) / 8
h_1	3	1, -1/2

Symmetric Boundary Extensions

- Consider a 1-D signal of length N_0. On the analysis side:
  - For transformation by WSS filters, the signal is extended to a whole-sample symmetric signal of length $2N_0$ -2 and periodized.
  - For HS-type filters, it is extended to a HSS signal of length $2N_0$ and periodized.
- In each case, the filters are extended with zeros to length N and applied by N-periodic circular convolution.
- The type of symmetric boundary extensions required for the synthesis side depend on the filter symmetries and the odd or even size of the signal.

Symmetric DWT Extensions



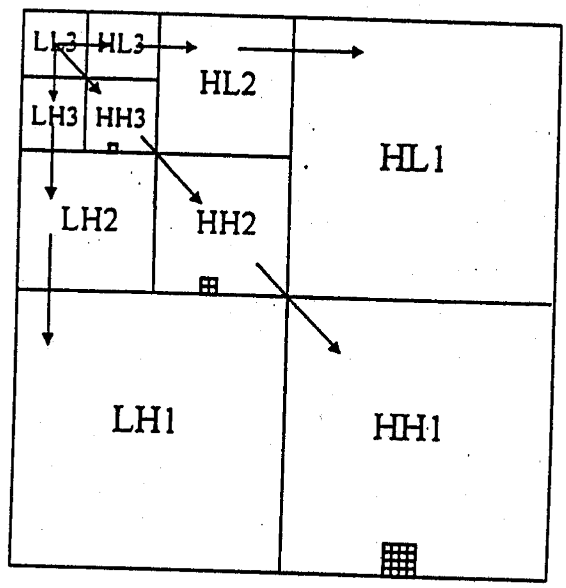
DWT Complexity Issues

- The complexity of the DWT depends on
  - Filter sizes,
  - Floating point vs. integer filters
- Except for a few special cases, e.g., the (5,3) integer filter, the DWT is generally more computationally complex (~2X to 3X) than the block-based DCT.
- As a full-frame transform, the DWT also requires significantly more memory than the DCT. However, line-based implementations can reduce the memory requirements.

Embedded Zerotree Wavelet (EZW) Coder

- The image is first transformed into wavelet coefficients by using a discrete wavelet transform (DWT).
- A wavelet coefficient is called **insignificant** with respect to a threshold *T* if its magnitude is less than *T*.
- The process of **zerotree** coding is based on the assumption that if a wavelet coefficient at a coarser scale (called a parent) is insignificant with respect to a threshold *T*, then all the wavelet coefficients of the same orientation in the same spatial location at finer scales (called the descendants) are also likely to be insignificant with respect to *T*.

Zerotrees



Embedded Zerotree Coding Procedure

- The symbols to be encoded at the first pass of the EZW method are generated in the following way:
  - An initial quantization threshold is selected (usually about half the magnitude of the largest wavelet coefficient).
  - The coefficients in each band are scanned in a raster order, by starting at the lowest frequency band, scanning the bands from left to right and top to bottom, and then moving on to the next scale and repeating the process.
  - Each wavelet coefficient is compared to the threshold T, resulting in one of four possibilities:

Embedded Zerotree Coding Procedure

- If the coefficient is significant and positive, the symbol encoded is (POS).
- If the coefficient is significant and negative, the symbol encoded is (NEG).
- If the coefficient is insignificant, but at least one of its descendants is significant, an *isolated zero* (IZ) is coded.
- If the coefficient is insignificant, its parent is significant, and all of its descendants are also insignificant, the symbol encoded is a *zerotree root* (ZTR).

- An arithmetic coder is used to encode these four symbols.

Embedded Coding

- To perform **embedded encoding**, successive approximation quantization is applied where both the significance of a coefficient and its magnitude are further refined by applying a sequence of thresholds $\{T_n\}$, such that $T_i = T_{i-1}/2$.
- The advantage of embedded encoding is that if the bit stream is truncated at any given point, an optimal image at that file size can be reconstructed.
- In order to accomplish this, two separate lists of wavelet coefficients are maintained, and at each pass, which corresponds to a given threshold T_j, these lists are updated:

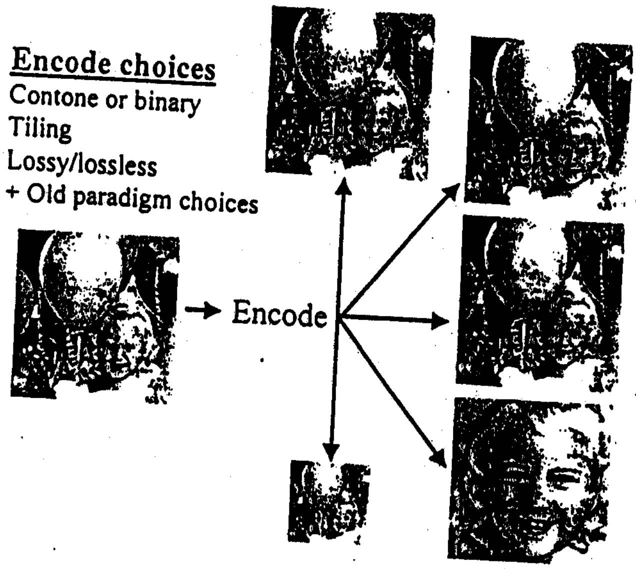
Dominant And Subordinate Lists

- A ~~dominant~~ insignificant list, containing the location of those coefficients that have so far been found to be insignificant. At each pass of this list, its coefficients are compared to the current threshold T_j to determine their significance and, if significant, their sign. This information is encoded with a procedure similar to the first pass. Note that the descendants of a zerotree can be skipped and need not be coded.
- A ~~subordinate~~ refinement list, containing the magnitudes of those coefficients that have been found to be significant. At each pass of this list, the magnitudes of the coefficients are refined to an additional bit of precision using two symbols. This is similar to **bit plane encoding** of the magnitudes.
- Arithmetic coding is used to encode all symbols.

The Emerging JPEG-2000 Standard

New Compression Paradigm

Encode choices
- Contone or binary
- Tiling
- Lossy/lossless
- + Old paradigm choices



Decode choices
- Image resolution
- Image fidelity
- Region-of-interest
- Fixed size
- Components
- Lossless/lossy

20

JPEG 2000 Objectives

- Advanced standardized image coding system to serve applications into the next millenium
- Address areas where current standards fail to produce the best quality or performance
- Provide capabilities to markets that currently do not use compression
- Provide an open system approach to imaging applications

82

JPEG-2000: Requirements And Profiles

- Internet applications (World Wide Web imagery)
  - Progressive in quality and resolution, fast decode
- Mobile applications
  - Error resilience, low power, progressive decoding
- Electronic commerce
  - Image security, digital watermarking
- Digital photography
  - Low complexity, compression efficiency

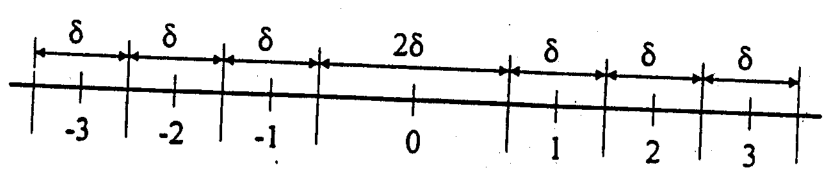
JPEG-2000: Requirements And Profiles

- Hardcopy color facsimile, printing and scanning
  - Compression efficiency, strip or tile processing
- Digital library/archive applications
  - Metadata, content management
- Remote sensing
  - Multiple components, fast encoding, region of interest
- Medical applications
  - Region of interest coding, lossy to lossless

JPEG 2000 Features

- Improved compression efficiency (estimated 30% depending on the image size and bit rate)
- Lossy to lossless
- Multiple resolution
- Embedded bit stream (progressive decoding)
- Region of interest coding (ROI)
- Error resilience
- Bit stream syntax (proposed by DIG 2000)

Deadzone Scalar Quantization

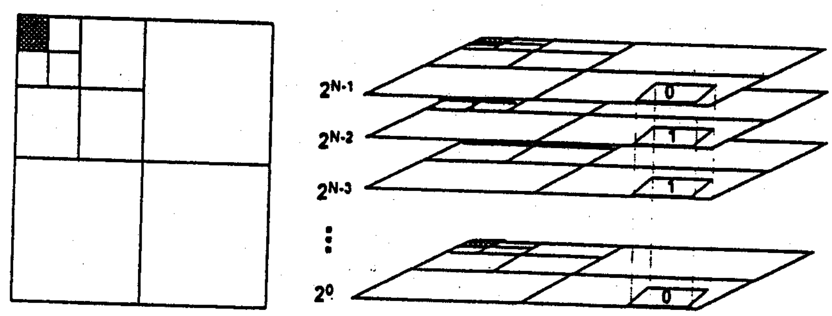


Consider the wavelet coefficient $s[m,n]$, with a sign s and a magnitude $|s[m,n]|$. Discarding n bit planes of $s[m,n]$ is equivalent to a deadzone quantization with a step size $\delta=2^n$. A dead-zone quantization is one in which the central dead-zone is twice as large as the step size δ.

Magnitude of quantizer index $= \left\lfloor \frac{|s[m,n]|}{\delta} \right\rfloor$, Sign $= s$

Bit-Plane Representation Of The Subbands

Implicit scalar quantization is similar to the truncation of an embedded code stream where the transform coefficients are bit plane by bit plane encoded in an embedded fashion.



DWT Filters In JPEG 2000 VM

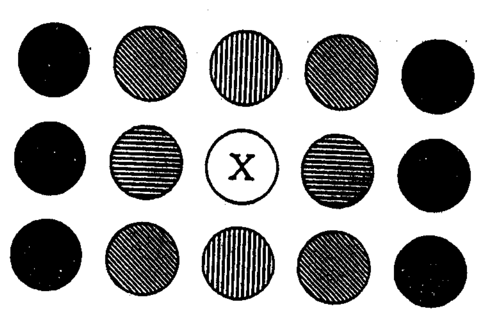
- Floating point filter: (9,7), (10,18)
- Integer filters: CRF(13,7), (5,3), (2,10), etc.
- Default integer filter for lossy coding: CRF (13,7)
- Default integer filter for lossless coding: (5,3), i.,e.,
  - H_0: (-1 2 6 2 -1) / 8
  - H_1: (-1 2 -1) / 2
- The current VM supports user-defined arbitrary size filters in addition to arbitrary wavelet decomposition trees.

Binary Arithmetic Coder

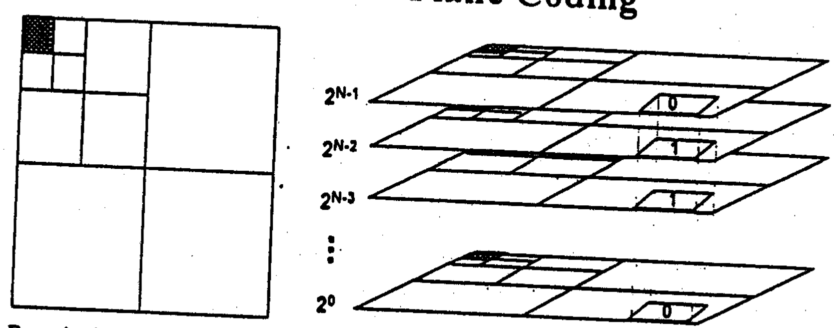
- The binary symbol (i.e., the bit plane value) in each context is coded using arithmetic coding.
- The choice of the specific binary arithmetic coder has not been finalized.
- Depending on the modeler and the encoder, each bit plane is coded into several *coding units*, e.g., in the current VM:
  - Predicted significance (PS)
  - Refinement (REF)
  - Predicted non-significance (PN)

Symbol Modeling And Coding

- In the current VM, the binary value at a given location in a bit plane of a subband is modeled by a context formed from its neighbors.



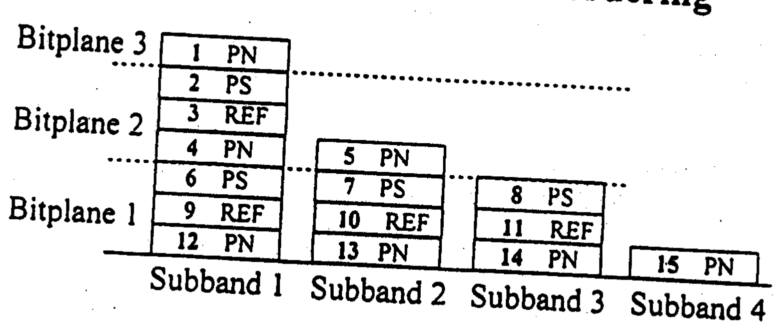
Scalable Bit-Plane Coding



- Resolution Scalability
  - Send quantized data in order of increasing scale
- SNR Scalability
  - Send bits in order that minimizes RMS Error

By the proper tagging of the encoded bitstream, resolution scalability, SNR scalability, or any other prioritization (up to lossless for integer wavelets) of the encoded data can be obtained.

Example Of Bit-Plane Reordering



- Progressive by quality (embedded bit stream)
  - 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15
- Progressive by resolution:
  - 1, 2, 3, 4, 6, 9, 12, 5, 7, 10, 13, 8, 11, 14, 15

Visual Frequency Weighting

- Two modes of visual weighting (VW) enable the system designers to take advantage of HVS properties:
  - **Fixed VW**: The weights are chosen according to the final viewing condition. Is implemented by scaling the quantization step size in a given subband.
  - **Visual progressive coding**: Visual weights change during the embedded encoding/decoding process. Is implemented using arbitrary reordering of the bit planes.
- The resulting MSE is usually higher than without VW, but the subjective quality is significantly improved.
- Design of the CSF weights is an encoder issue.

Region Of Interest (ROI) Coding

- Allows selected parts of an image to be coded with higher quality.
- ROI can have arbitrary shape. The location and the shape of ROI is transmitted to the receiver.
- Due to the nature of the wavelet decomposition, the ROI degradation is graceful.
- ROI can be specified either in the beginning or during the encoding process (e.g., by the receiver who requests the lossless transmission of a certain image region).
- The current VM supports two methods of ROI encoding.

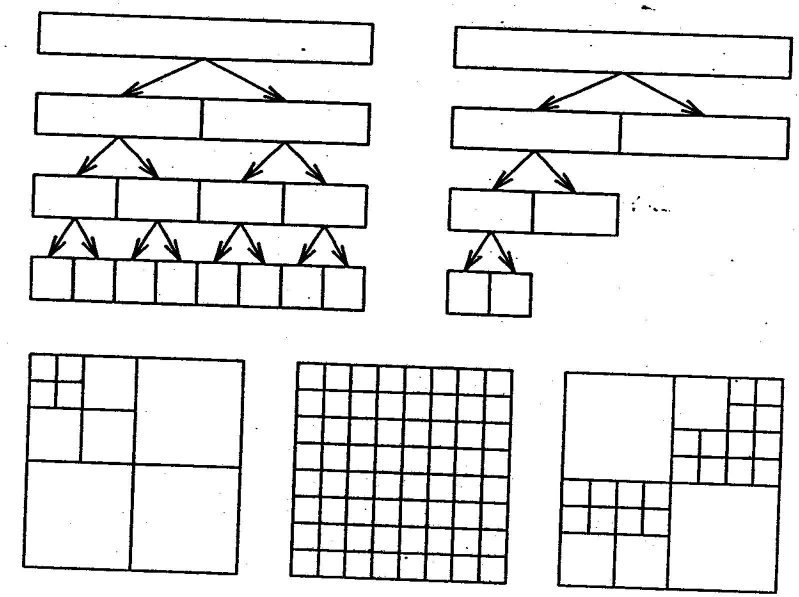
Figure 7. Edge detected from the local maxima of the wavelet transform modulus, at the scale 2^2.

Figure 8. Edge curves selected with a thresholding on their length and on the average value of the wavelet transform modulus, at the scale 2^2. The length and modulus thresholds are twice larger than in Figure 5.

FIGURE 14. The top left of (a), (b) and (c) gives the original image of 256 by 256 pixels. The top right is the reconstructed image from the coded multiscale representation. Image (a) requires 0.30 bits per pixel, image (b) 0.24 bits per pixel and image (c) 0.20 bits per pixel. The bottom right gives the edge curves that are encoded at the scale 2^2. The bottom left shows the reduced image that carries the remaining low-frequency information.

FIGURE 14 c.

Wavelet Packets und der "best basis Algorithmus"



- Best Basis Algorithmus: eine Möglichkeit, gute Zerlegungen zu finden.
- **Entropy**: Let $X = (x_0, x_1, x_2, \ldots)$ be the sequence of coefficients. The Shannon — Weaver entropy of X is $H(X) = -\Sigma_j p_j \log p_j$, where $p_j = \frac{|x_j|^2}{||x||^2}$. An additive analogon to Shannon — Weaver entropy is $\lambda(X) = -\Sigma_j |x_j|^2 \log |x_j|^2$. The relation $H(X) = ||x||^{-2}\lambda(X) + \log ||x||^2$ insures that minimizing the latter minimizes the former.

Ein Vergleich von Wavelet und JPEG basierten selektiven Methoden im Bereich der medizinischen Bildkompression

Alfred Bruckmann & Andreas Uhl

RIST++, Universität Salzburg

(1) Hintergrund

Bei bildgebenden Verfahren in der Medizin (z.B. CT, MRI, Ultraschall, etc.) setzt sich die Verwendung von digitalen Methoden mehr und mehr durch. Die dadurch anfallenden Datenmengen in einem größerem Klinikum liegen im Bereich von einigen Terabyte pro Tag. Auch für Telemedizinische Anwendungen ist die Bandbreite der verfügbaren Netzwerktechnologien für die anfallenden Datenmengen zu klein.
Die Kompression von solchen Bilddaten hilft, die Datenmengen zu reduzieren. Es gibt in der klassischen Bildverarbeitung 2 grundsätzliche Methoden:

- **Lossless Kompression**: das dekomprimierte Bild ist mit dem ursprünglichen Bild identisch (z.B. .gif Format). Die Datenreduktion ist bei einem 512×512 Pixel Bild mit 8 bit Farbtiefe auf ca. Faktor 3 beschränkt.
- **Lossy Kompression**: das dekomprimierte Bild ist eine Approximation des ursprünglichen Bildes, bei starker Kompression kommt es zur Entstehung von sichtbaren Artefakten (z.B. .jpg Format). Die Datenreduktion kann bei einem Bild wie oben weit über Faktor 100 gehen.

(2) Problem

Die enormen Datenmengen die in der medizinischen Bildgebung anfallen verlangen eigentlich den Einsatz von Lossy Kompression. Dies wird jedoch von vielen MedizinerInnen aus Bedenken wegen möglicher Fehldiagnosen auf Grund von Artefakten abgelehnt.

Darüberhinaus sind in vielen medizinischen Bereichen Lossless Verfahren vom Gesetzgeber vorgeschrieben.

Die vorliegende Arbeit löst diesen scheinbaren Widerspruch mit der Entwicklung eines neuen Kombinationsverfahren auf: der **Selektiven Bildkompression** (SLIC).

(3) Lösunganſatz: Selektive Bildkompression

Selektive Kompression vereinigt Lossless und Lossy Verfahren in einer Methode nach dem Motto "Best of both worlds". Wichtige Bildinformation wird Lossless komprimiert, unwichtige Hintergrund- oder Nebeninformation Lossy.

Bei dem Verfahren gibt es zwei Vorgehensmöglichkeiten:

1. Automatische Klassifikation in "wichtige" und "unwichtige" Bildteile (z.B. können Microcalcifications in Mammographien effizient automatisiert erkannt werden).
2. Ein(e) RadiologIn legt interaktiv "wichtige" und "unwichtige" Bildteile am Bildschirm fest. In unserer Arbeit betrachten wir diesen Fall.

(4) Mögliche Anwendungsszenarien: Telemedizin

Ein(e) MedizinerIn in einem Provinzkrankenhaus ist per Modem mit der Radiologischen Abteilung eines Klinikzentrums verbunden und möchte dort Fachleute in einem problematischen Fall konsultieren. "Verdächtige" Strukturen werden am Schirm markiert und Lossless komprimiert, der Rest Lossy. Dann werden die Daten geschickt.

Auf dem Bildschirm im Klinikzentrum erscheinen nun die diagnostisch relevanten Bildteile ohne Veränderung, der unwichtige Hintergrund bleibt als (wenn auf leicht veränderte) Umgebung erhalten.

Resultat: die diagnostische Genauigkeit wird beibehalten, die Datentransferzeit zwischen Teilnehmern wird erheblich reduziert (das ist besonders wichtig bei interaktiven Prozessen).

(5) Mögliche Anwendungsszenarien: Bildarchivierung und Retrieval

Ein(e) MedizinerIn befundet und archiviert medizinisches Bildmaterial. Die Bildteile auf die sich die Diagnose stützt werden markiert und Lossless gespeichert. Das restliche Datenmaterial wird Lossy gespeichert um die Umgebung adaquat, jedoch nicht völlig authentisch darzustellen.

Weiters kann das Residual (Differenzbild) zusätzlich gespeichert werden. Dies ermöglicht im Fall einer Datenbankabfrage eine schnelle erste Darstellung der Originalbildes, bei Bedarf kann es aber sogar völlig rekonstruiert werden.

Resultat: erhebliche Reduzierung der Datenmenge bei gleichzeitigem Einhalten der juristischen Auflagen.

JPEG Basierte Methoden

Algorithmus J1

- Die Form und Position der Region(en) wird als (komprimiertes) binäres Bild gespeichert ("importance map").
- Der Inhalt wird mit einem Arithmetischen Coder (verlustfrei) komprimiert.
- Der unwichtige Teil des Bildes wird mit einem JPEG Coder (verlustbehaftet) komprimiert.

Algorithmus J2

- Im Gegensatz zur ersten Methode wird hierbei das ganze Bild mit einem JPEG Coder komprimiert.
- Wiederherstellung des Bildes (mit einem Verlust an Qualität)
- Berechnung eines Fehlerbildes beschränkt auf den signifikanten Bereich.
- Dieses Fehlerbild wird zusammen mit der "importance map" verlustfrei gespeichert.

Wavelet Basierte Methoden

Algorithmus W1

- Die Form und Position der Region(en) wird als (komprimiertes) binäres Bild gespeichert ("importance map").
- Der Inhalt wird mit einem Arithmetischen Coder (verlustfrei) komprimiert.
- Der unwichtige Teil des Bildes wird mit einem Wavelet Coder (verlustbehaftet) komprimiert.

Algorithmus W2

- Im Gegensatz zur ersten Methode wird hierbei das ganze Bild mit einem Wavelet Coder komprimiert.
- Wiederherstellung des Bildes (mit einem Verlust an Qualität)
- Berechnung eines Fehlerbildes beschränkt auf den signifikanten Bereich.
- Dieses Fehlerbild wird zusammen mit der "importance map" verlustfrei gespeichert.

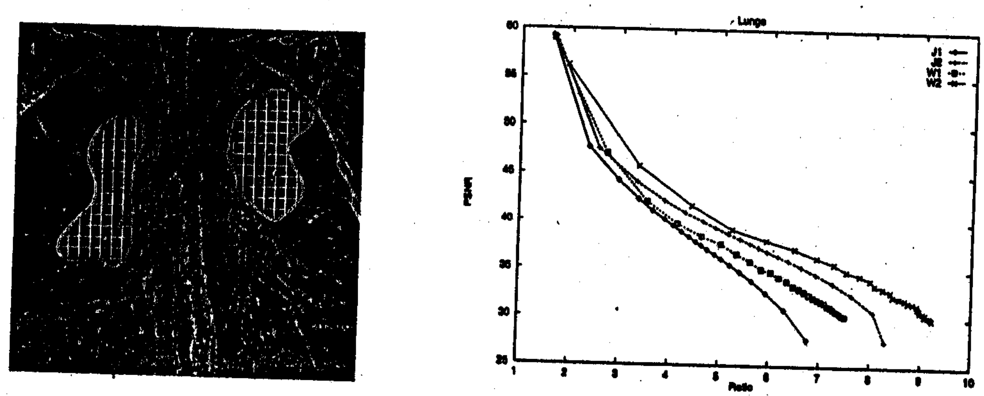


Figure 1: Selektion und Kompressionsleistung für ein Lungen CT

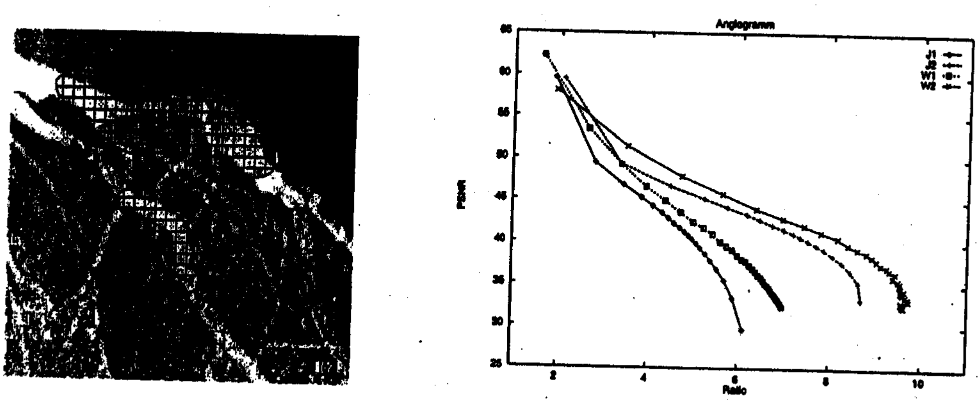
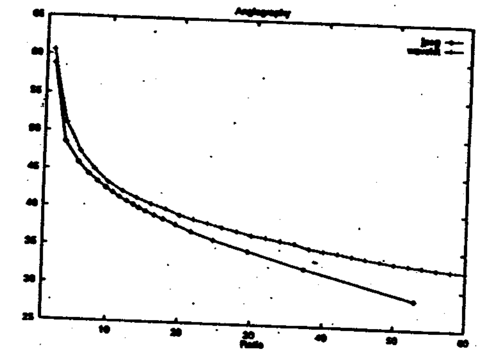


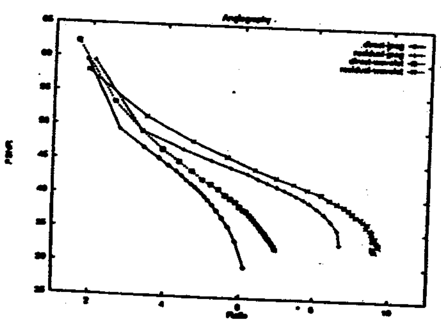
Figure 2: Selektion und Kompressionsleistung für ein Angiogramm

(a)

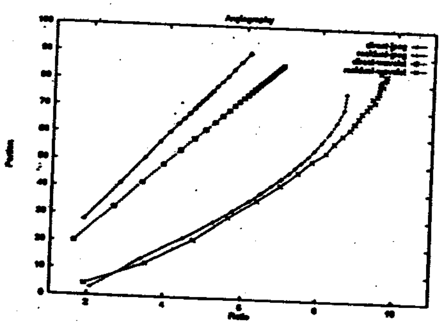


(b)

Figure 5: Digital angiography (8bpp, 512 × 512 pixels) with RoI consisting of two regions (a) and rate-distortion performance of the basic compression algorithms applied to the entire image (b).

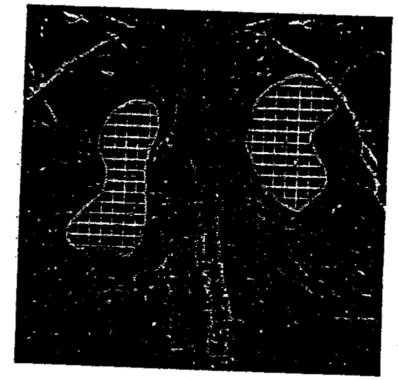


(a)

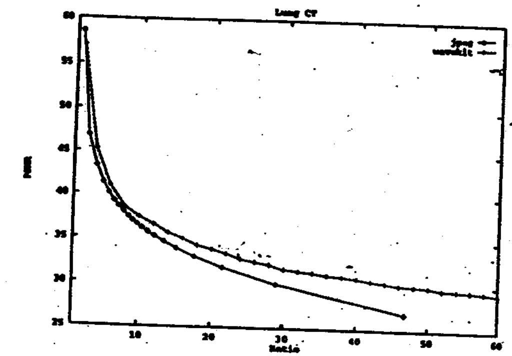


(b)

Figure 6: Comparison of rate-distortion performance of different selective compression algorithms applied to the angiography (see Figure 5.a) (a) and comparison of the corresponding portions of the RoI data in percent (b).

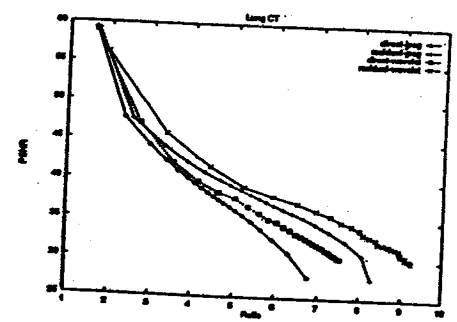


(a)

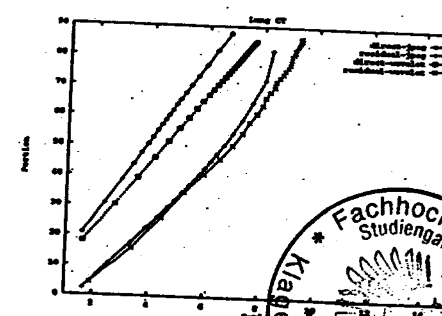


(b)

Figure 3: Test image (8bpp, 512 × 512 pixels Lung CT) with RoI consisting of two regions (a) and rate-distortion performance of the basic compression algorithms applied to the entire image (b).



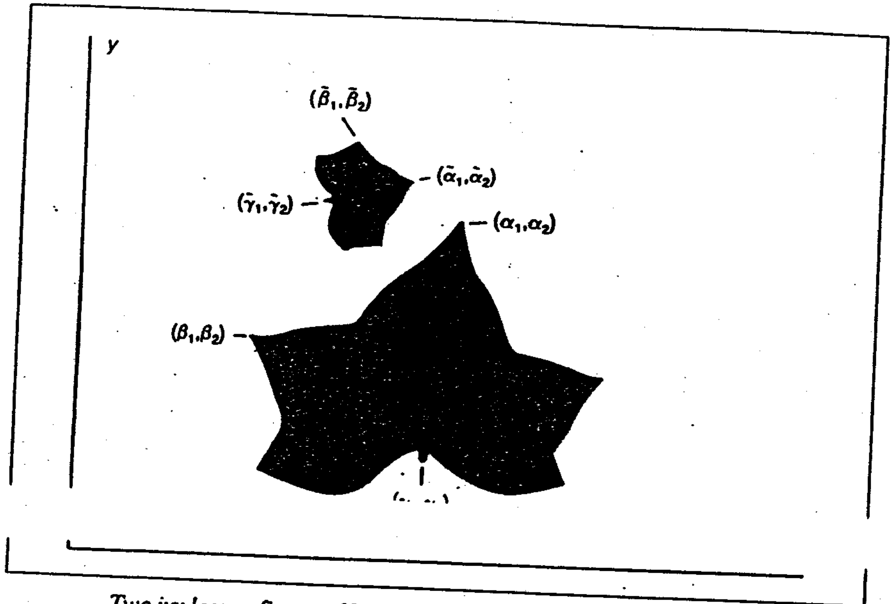
(a)



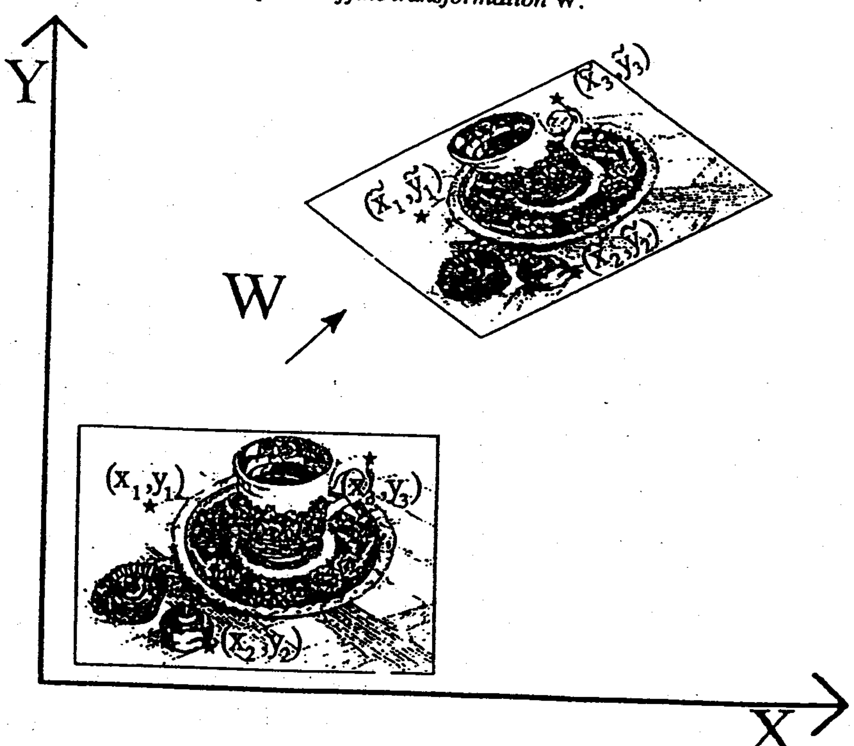
(b)

Figure 4: Comparison of rate-distortion performance of different selective compression algorithms applied to the Lung CT (see Figure 3.a) (a) and comparison of the co

Affine Transformation



Two ivy leaves fix an affine transformation W.



An affine transformation is determined

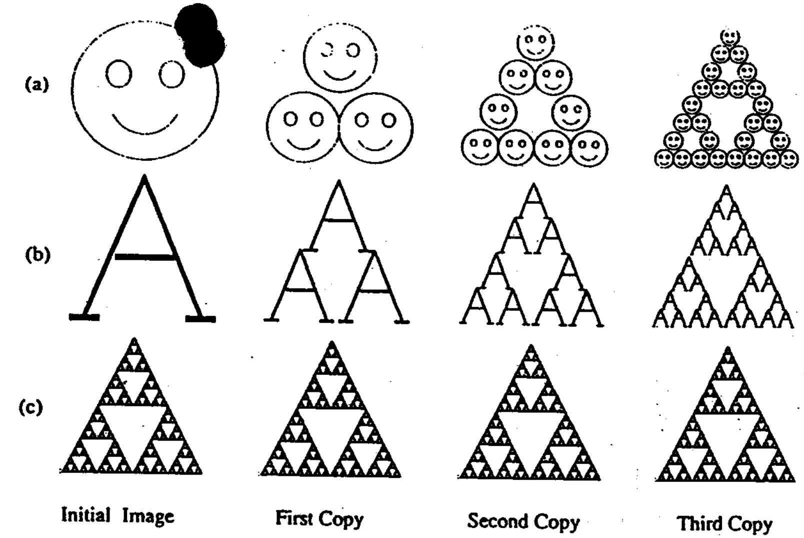


Figure 1.2: The first three copies generated on the copying machine of Figure 1.1.

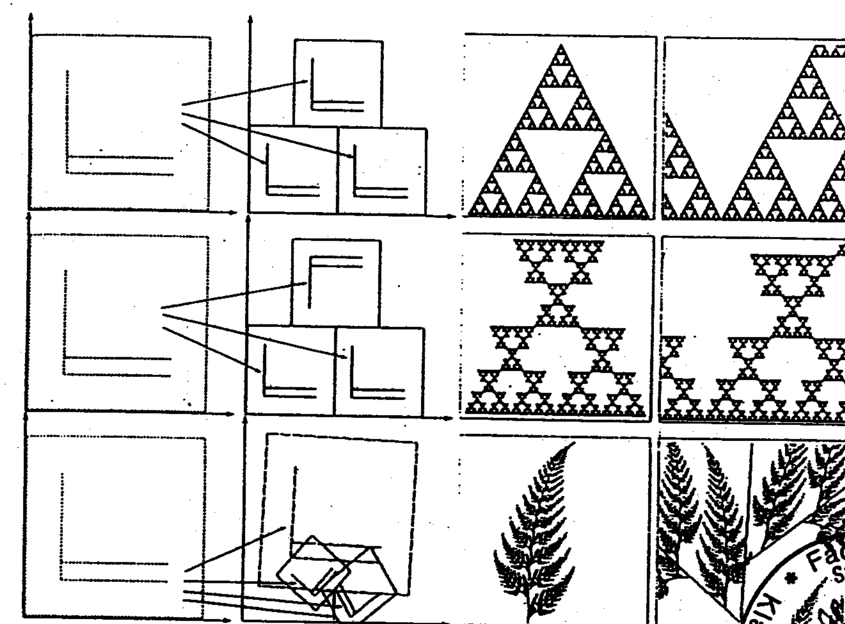
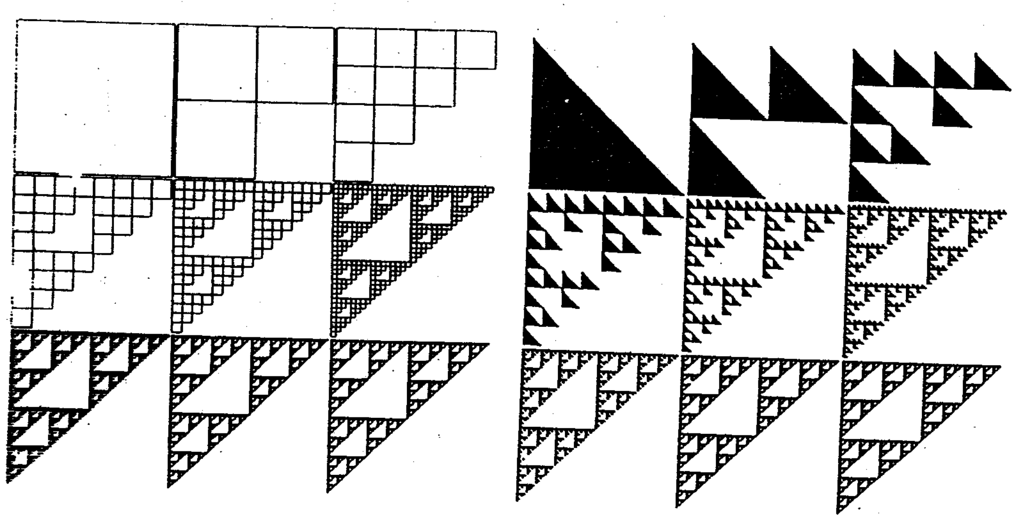


Figure 1.3: Transformations, their attractor, and a zoom on the attractors

IFS: $\{X; w_n; n = 1, 2, 3\}$ mit den folgenden w_i's:

$$w_1\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$
$$w_2\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{2} \end{pmatrix},$$
$$w_3\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}.$$



Fractal image compression

1. Partition image $\vec{x}$ into non-overlapping **ranges** $\vec{r_i}$ with $\vec{r_i} \cap \vec{r_j} = 0$.
   The union of all the ranges is the image $\vec{x}$ itself:
   $\vec{x} = \bigcup \vec{r_i}$.

2. Determine a set of blocks **(domain pool)** $\{\vec{d_i}\}$.
   The union of all domains is a subset of the image $\vec{x}$, $\bigcup \vec{d_i} \subseteq \vec{x}$.

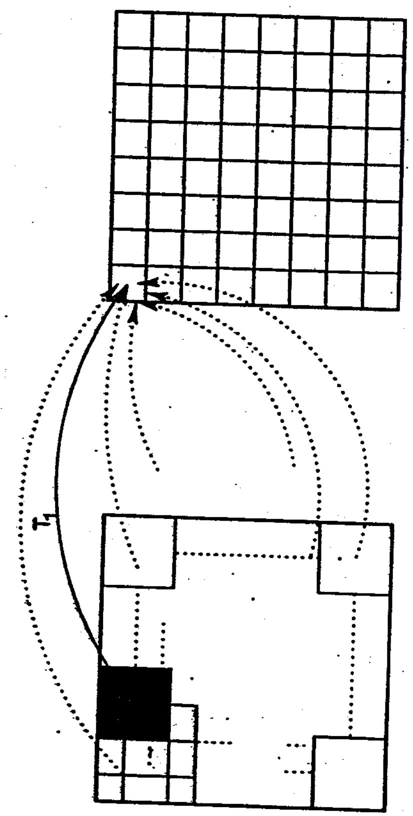
3. Define an operator $T : \{d_j\} \longrightarrow \{r_i\}$.

$$T = \bigcup_i T_i \quad \text{with} \quad T_i : d_j \longrightarrow r_i,$$

$$\text{so that} \quad d_2(\vec{x}, T(\vec{x})) < \Delta$$

4. Store the coefficients of T into the fractal file.

For each range search for the "most similar" domain



Define an operator $T : \{d_j\} \longrightarrow \{r_i\}$.

$$T = \bigcup_{i=1}^{\#r} T_i \quad \text{with} \quad T_i : d_j \longrightarrow r_i,$$

$$\text{so that} \quad d_2(\vec{x}, T(\vec{x})) < \Delta$$

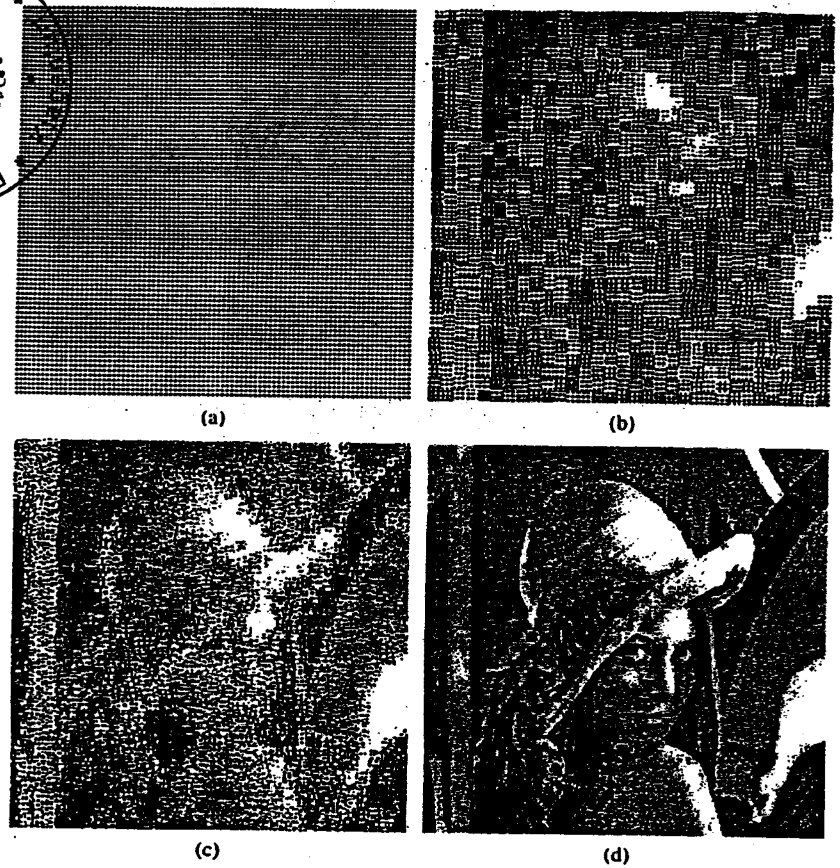
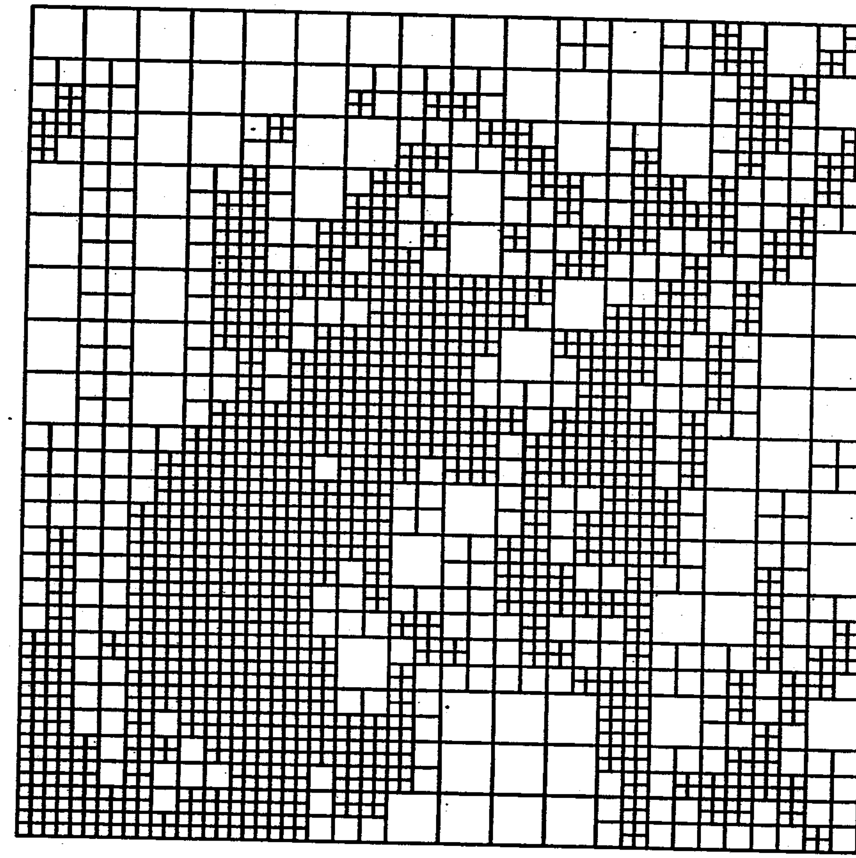
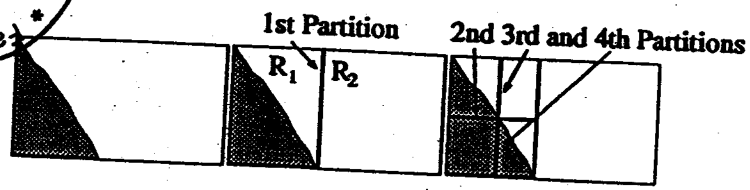


Figure 1.10: The initial image (a), and the first (b), second (c), and tenth (d) iterates of the encoding transformations.

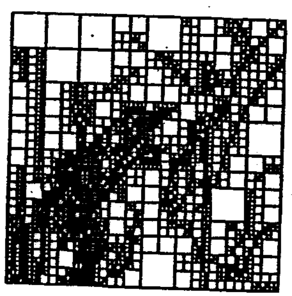


Partition of the image Lena with an initial range size of 16×16 and a minimal range size of 4×4 pixels. The quality criteria (rms) is 8.0.

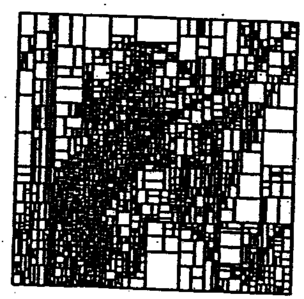
HV-Partitioning



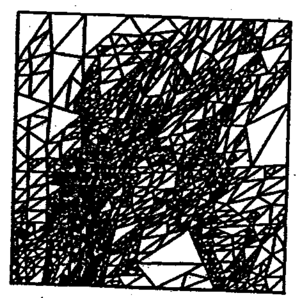
Diverse Unterteilungen



Quadtree
5008 Quadrate



HV
2310 Rechtecke



Triangulation
2954 Dreiecke

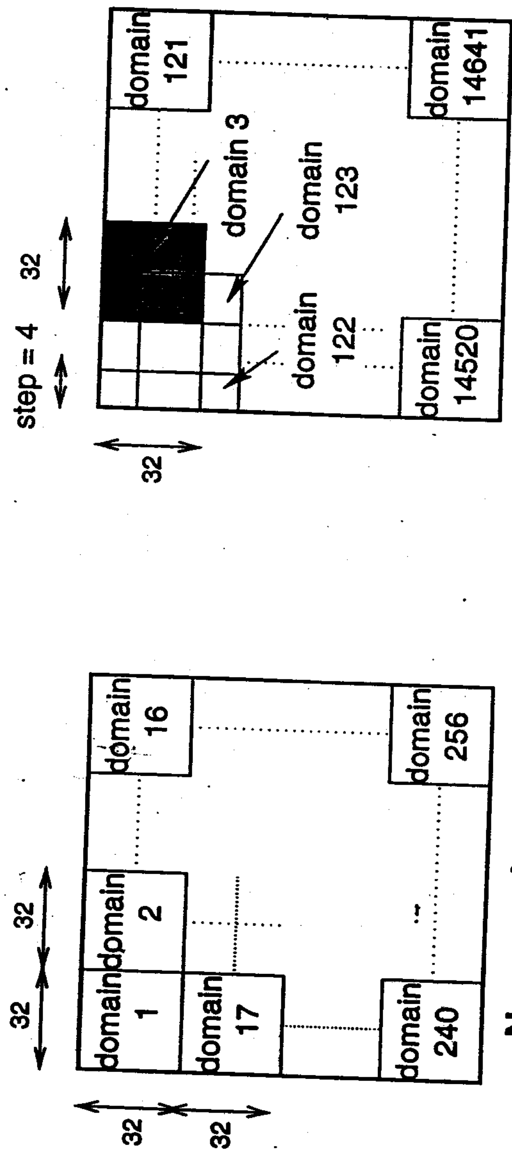
Encoding complexity

A complete domain pool of an image of size $n \times n$ with square domains of size $D \times D$ consists of $(n-D+1)^2$ domains. Each domain has 8 isometries. So each range must be compared with $8(n-D+1)^2$ domains.

Example:

- Image with size 256×256.
- Ranges with size 8×8 (= 1024 ranges).
- Domains with size 16×16 (= 504.008 domains).

So a full domain search for all ranges results in 516.104.192 tests (which includes rms-calculation).



Non-overlapping

Lattice with spacing 4

Domain pools of an image of size 512×512 with non-overlapping and overlapping domains.

Speedup techniques

- Efficient sequential search techniques:
  - Geometric searching techniques.
  - Discrete feature extraction techniques with
    * fixed predefined classes,
    * heuristic predefined classes, and
    * image adapted classes.
  - Continuous feature extraction techniques
    * 1D functional methods,
    * Feature vectors.
  - Mixed-type techniques.
- Parallelization (SIMD, MIMD, Vector, ...).

Grey-Value Classification

1. The subimage is divided into four quadrants. The pixel values in the quadrant i of block $\vec{r}$ are $r_1^i, \dots, r_n^i$ for $i = 1, 2, 3, 4$.
2. Then the sum of the grey values in a quadrant

$$A_i = \sum_{j=1}^{n} r_j^i$$

is computed for each quadrant. Then the subimage falls into one of the following classes:

Major Class 1: $A_1 \geq A_2 \geq A_3 \geq A_4$.
Major Class 2: $A_1 \geq A_2 \geq A_4 \geq A_3$.
Major Class 3: $A_1 \geq A_4 \geq A_2 \geq A_3$.

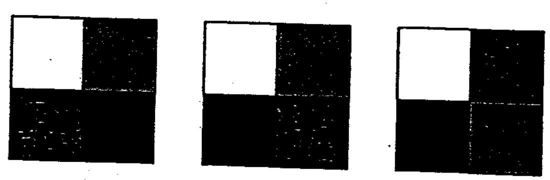


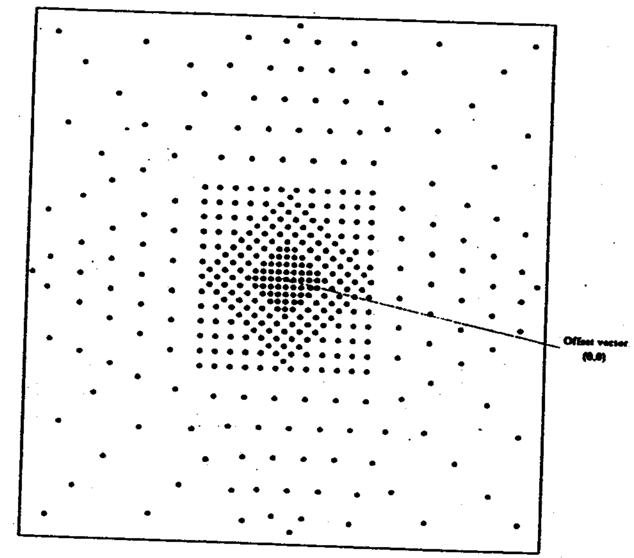
Figure 1: Major classes of the grey value classification

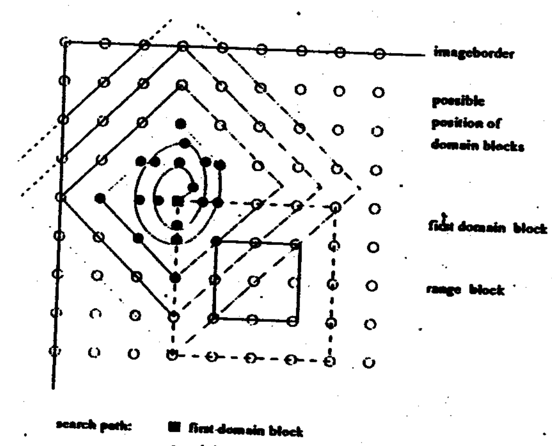
3. Then a variance

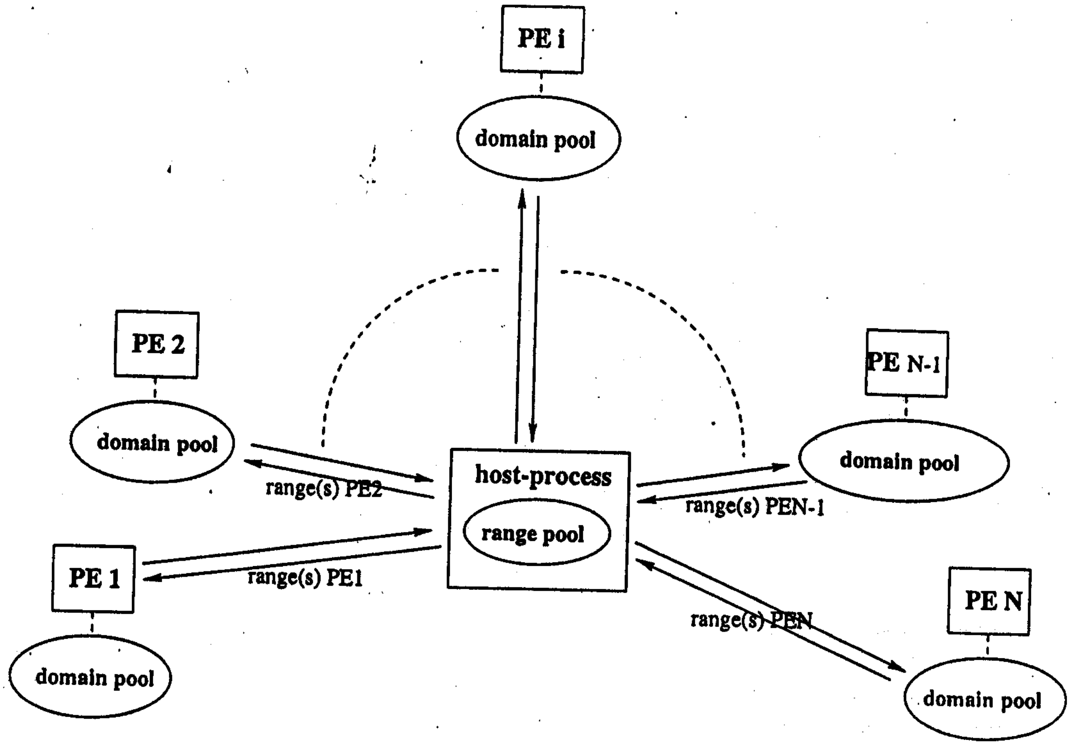
$$V_i = \sum_{j=1}^{n} (r_j^i)^2 - A_i^2$$

is computed for each quadrant.

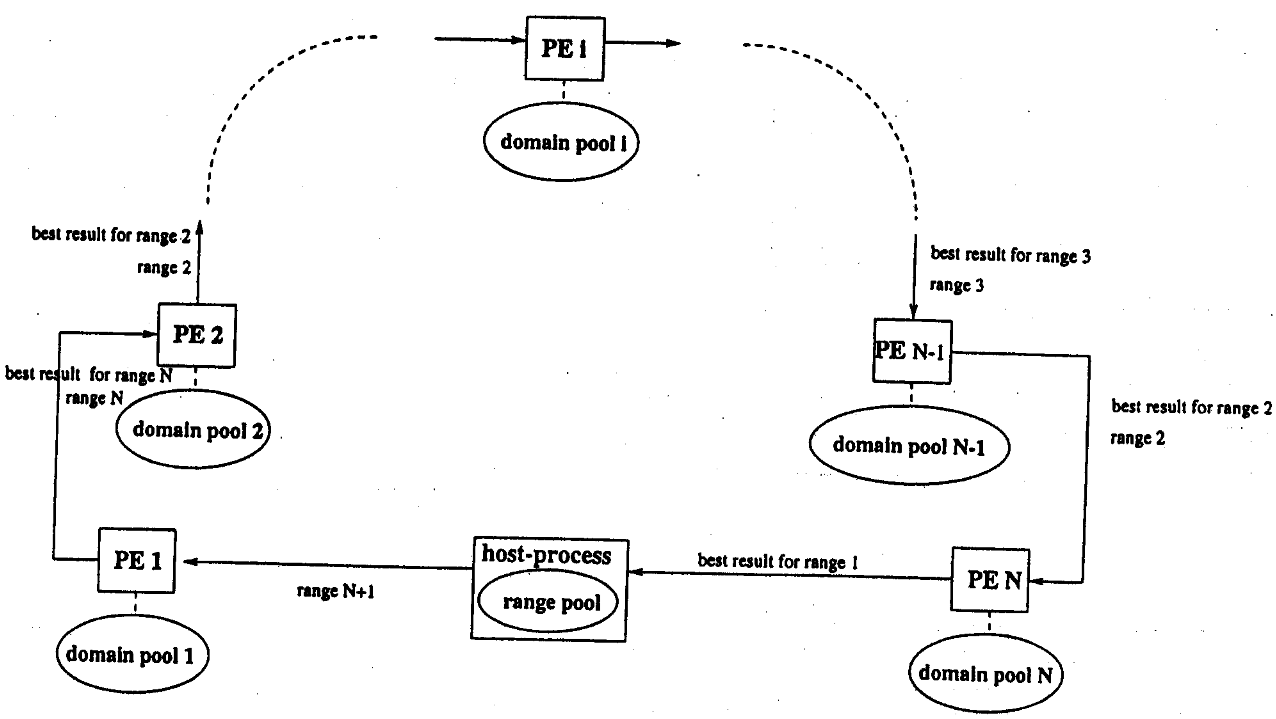
Geometric searching







Fractal Algorithm A



Fractal Algorithm B Pipe

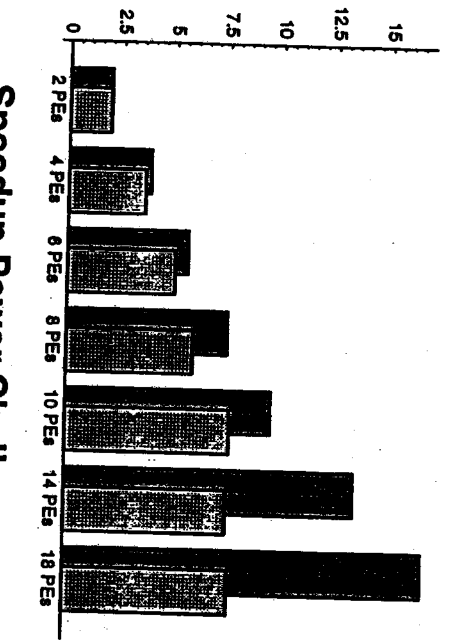
Computing Facilities

NOW: Network of workstations with 8 DEC AXP 3000/400 interconnected by a fibre distributed data interface (FDDI).

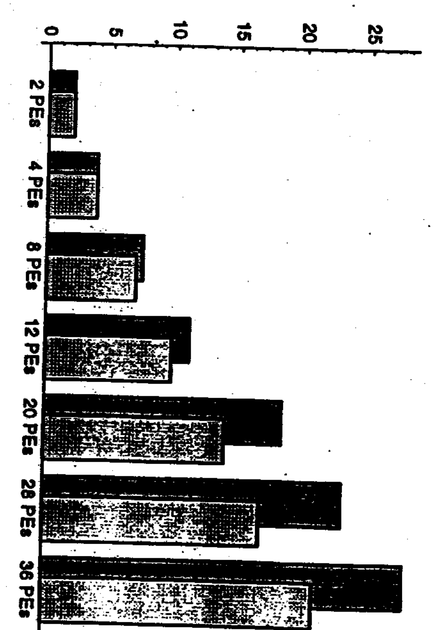
SGI Power Challenge GR: Shared memory machine based on the R10000 MIPS processor with 20 PEs (2 MB cache each) and 2.5 GB memory. (Located at the RIST++, University of Salzburg.)

Meiko CS-2: Distributed memory machine based on the Sun Super Sparc processor with 128 nodes, biggest partition: 64 nodes. (Located at the Vienna Center for Parallel Computing.)

Parsytec CC-48: Distributed memory machine based on the Motorola Power PC 604 with 48 nodes (8 IO-nodes and 40 PEs) and 3.0 GB memory. (Located at PC2, Paderborn Center for Parallel Computing.)



Speedup Power Challenge

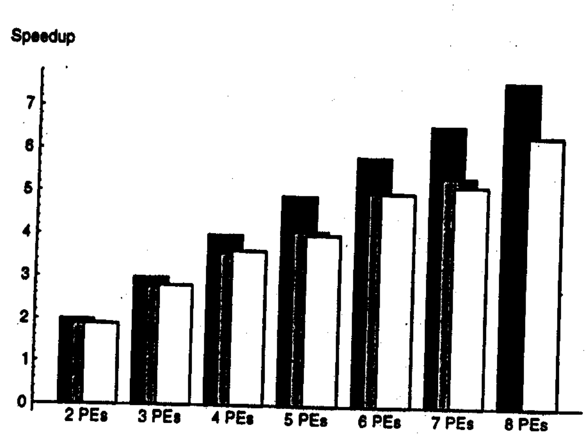


Speedup Parsytec CC-48

Dark: Algorithm A
Light: Algorithm B

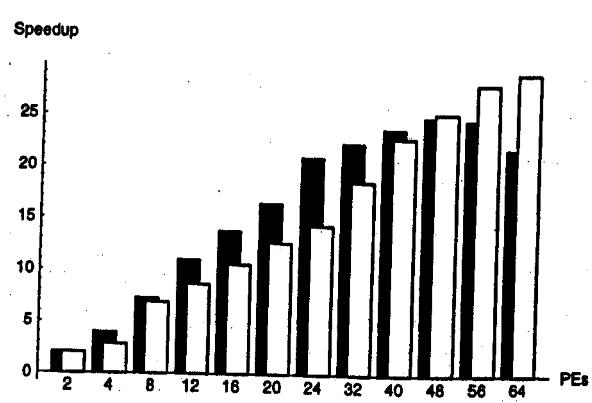
Load Balancing Strategies

1. Dynamic assignment of single ranges.
2. Dynamic assignment of blocks bigger than a range.
3. Static assignment of all ranges at the first quadtree-level.
4. Exact distribution on each quadtree-level.
5. A part of the range-pool (25 %, 50%, or 75%) is assigned at once. Remaining ranges are kept to fill load-imbalances.



Speedup NOW

Black: Algorithms A.
Gray: B Pipe with isolated host.
White: B Pipe with non-isolated host.



Speedup Meiko

Black: Algorithms A.
White: Algorithm B Pipe.

Combination: Sequential and Parallel Methods

Two possibilities:

- Sequential Methods and Class A Algorithms,
- Sequential Methods and Class B Algorithms.

Parallel algorithms of class A are very efficient and show nearly linear speedup in the order of the number of processors. They may include dynamic load balancing, and the number of PEs can be larger than with algorithms of class B before a bottleneck in the host-process occurs.

$\Longrightarrow$ Algorithms of class A are a good choice if the memory capacity of the PEs is sufficient.

Sequential Methods and Class B Algorithms: Fixed Distribution

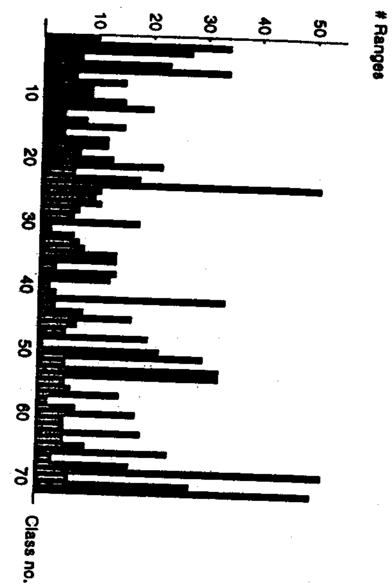
The uniform distribution of the domain-pool is done prior to any precalculations.

- Problem: the differences of the sizes of the classes among the different PEs might be large.
- Processing time for a given range might be very different on different PEs.
- Requirement for optimal speedup: uniform size distribution of the domain classes (pipelined approach) or identical size distribution of the domain classes (centralized approach).

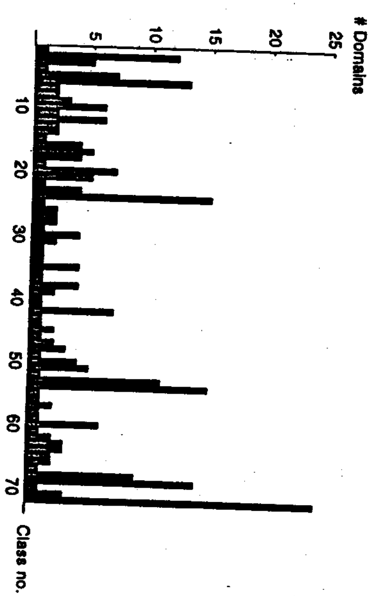
Sequential Methods and Class B Algorithms: Adaptive Distribution

We need: Knowledge about the class structure and class size.

- Not domains are distributed but classes (domains and ranges of one class).
- A class complexity is computed (= computation complexity).
- The total of all class complexities gives the overall complexity which has to be distributed evenly among the PEs.
- **Optimization Problem:** Given the number of PEs and the precalculated class complexities. Distribute the class complexities as uniformly as possible among the PEs (for each quadtreelevel).



Ranges



Domains

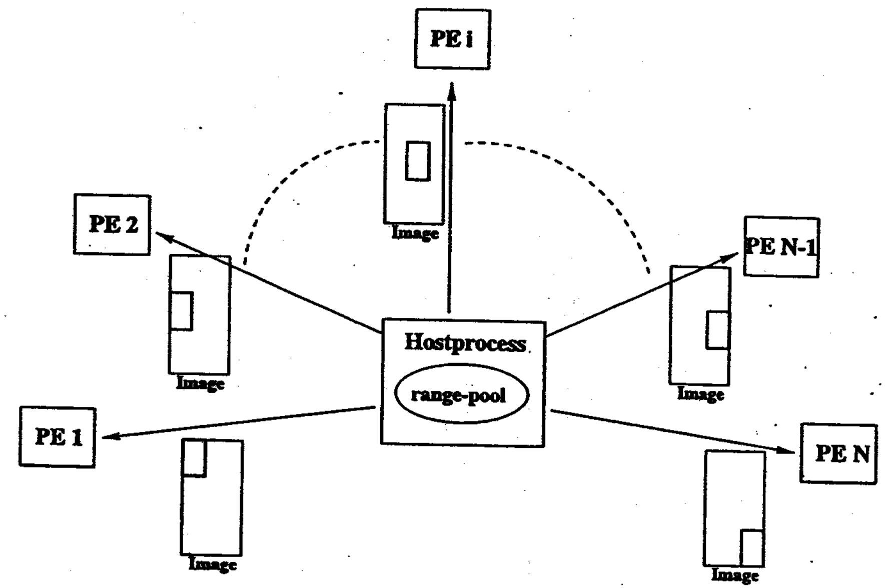
Image Lena 512 × 512
Number of blocks in a class. Calculated with the gray-value classification scheme.

Improvements of Algorithm A

Algorithm A can be improved concerning execution time through:

- Classification of domain-pool on PEs.

- "Localized domain-pools": The PEs use only a part of the domain-pool. So for a given range only that subdomain-pool is being searched which is located on the same PE.



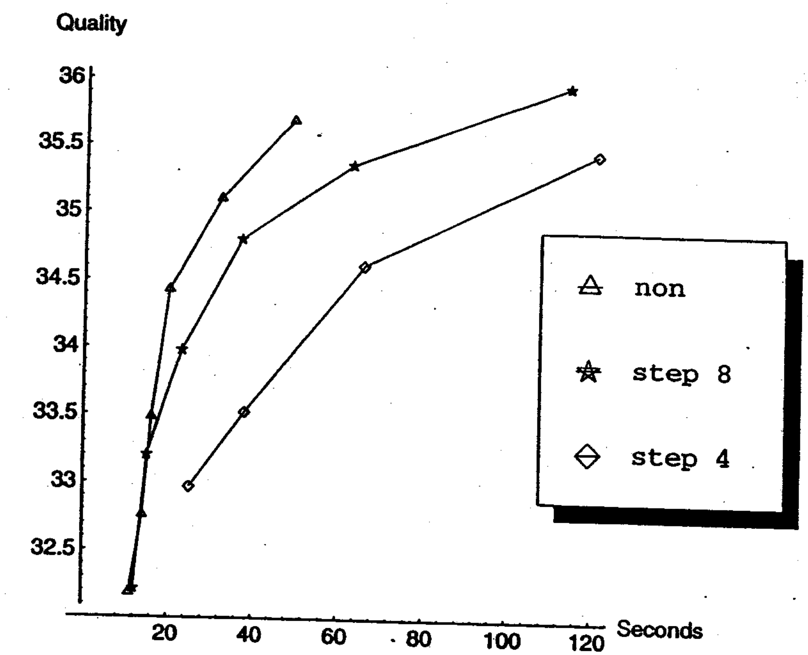
Each PE receives a part of the image to calculate its domain-pool. A redundancy factor k determines the size of the image part: Each PE uses a domain-pool of $\frac{1}{k}$ times the size of the original pool. If k is 0 each PE has a different pool (= the image parts do not overlap).

Localized Domain-pool

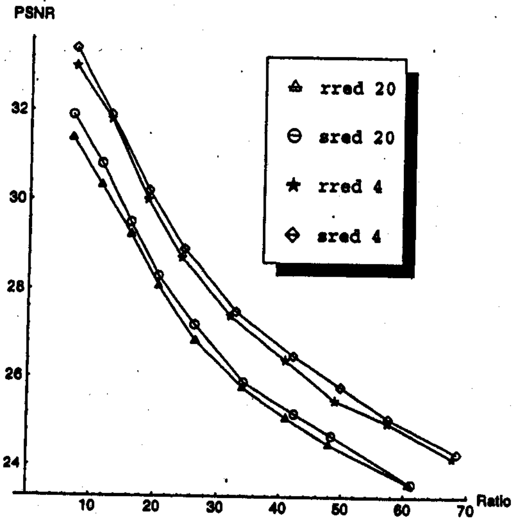
If we use a localized domain-pool the ranges can be assigned by the hostprocess in two different ways:

- **Dynamic random** assignment of ranges. This includes dynamic load-balancing where the location of the range is not taken into account.

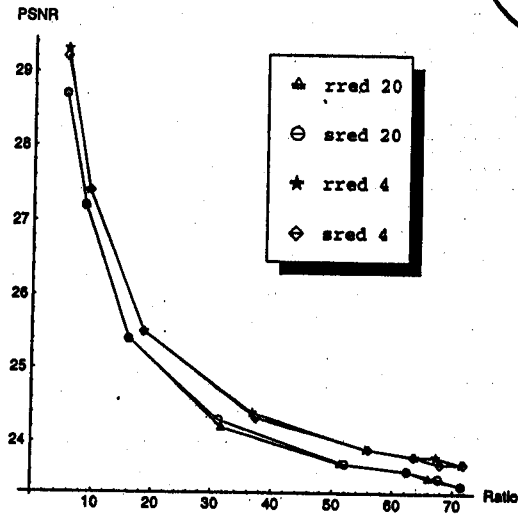
- **Static "geometric"** assignment of ranges. All the ranges are assigned at once. Each PE receives those ranges that are "close" to the location of its domains.



Quality performance of static range assignment methods with different domain-pools (non-overlapping, step 8, step 4) on the CC-48 (35 PEs), image Lena. The increasing time-demand is caused by a decreasing redundancy factor (divided by 2) in each measure point.



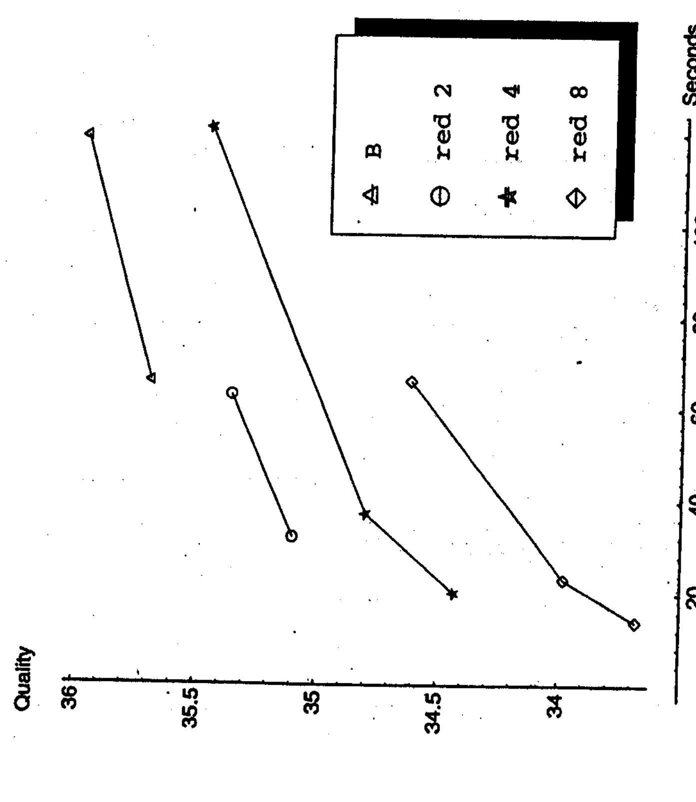
Lena



Satellite

Rate-distortion performance of static (sred) and random (rred) range assignment. The redundancy factors are 20 (sred 20, rred20) or 4 (sred4, rred4). The domain-pool is non-overlapping.

Results: Localized domainpools & Algorithm B



Comparison of different redundancy factors using 35 nodes of the CC-48.

Parallel Algorithms and Implementations

Previous Work: Especially on MIMD architecture. Two different types of algorithms exist: Parallelization via Ranges, and Parallelization via Domains.

Parallelization via Ranges: The complete image has to stored in each PE, to every PE a subset of the domain pool is assigned.

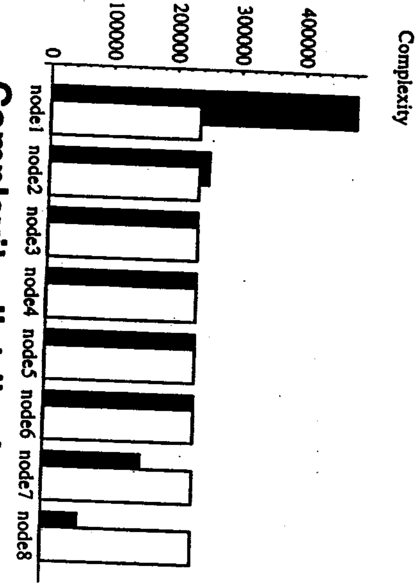
Parallelization via Domains: The domain pool is distributed evenly among the PEs.

Algorithms for SIMD: Due to memory constraints parallelization via ranges does not work, several questions arise: How can the domains be distributed in an efficient way ? How can the range-domain comparisons be organized ?

2-Arrays Approach: Domains are distributed evenly among the PEs, as well as the ranges, therefore two (virtual) arrays are mapped onto the PE–array. Each PE performs a range-domain comparison, then the array containing the ranges is shifted by one PE and again a comparison. The best result for each range is stored. For the next quadtree level a new distribution or some local balancing technique can be used.

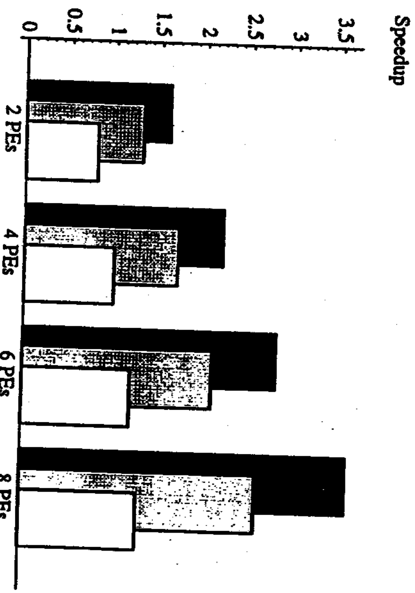
Pipelined Approach: The domains are distributed evenly among one row of the PE array, the ranges are distributed as in the 2-Arrays Approach. Especially useful for small domain pools. For the next quadtree level the same procedures are applied.

Broadcast Approach: domains are distributed evenly among PEs, ranges are broadcasted subsequently to the PEs, a global minimum function is used to find the best match; no load balancing problems.



Complexity distribution

Black: Adaptive without splitting.
White: Adaptive with splitting.



Speedup NOW

Black: Adaptive with splitting.
Gray: Adaptive without splitting.
White: Fixed distribution.

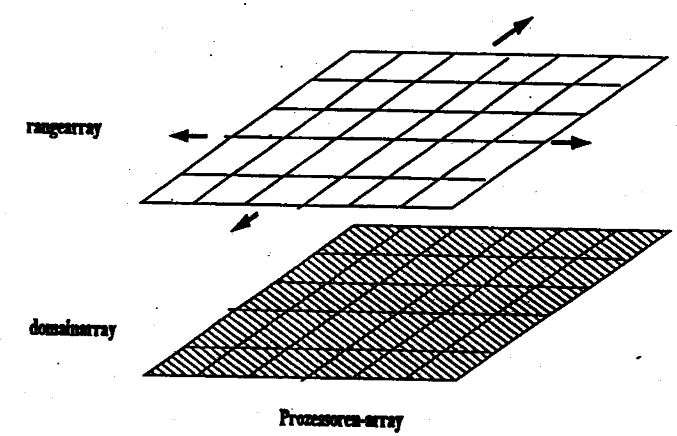


Figure 1: 2-array approach.

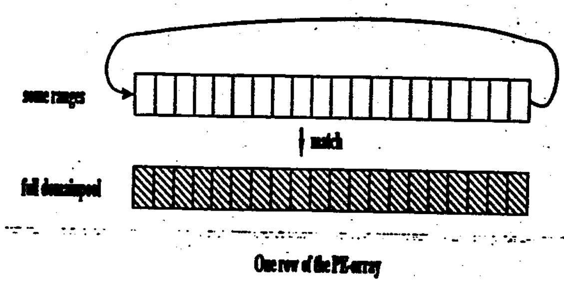


Figure 2: The pipeline approach.

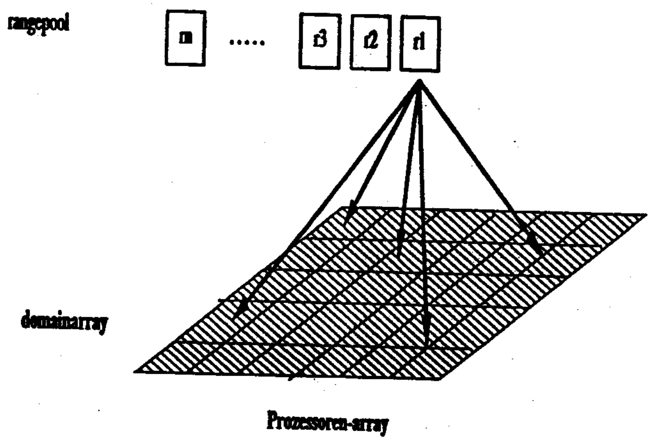
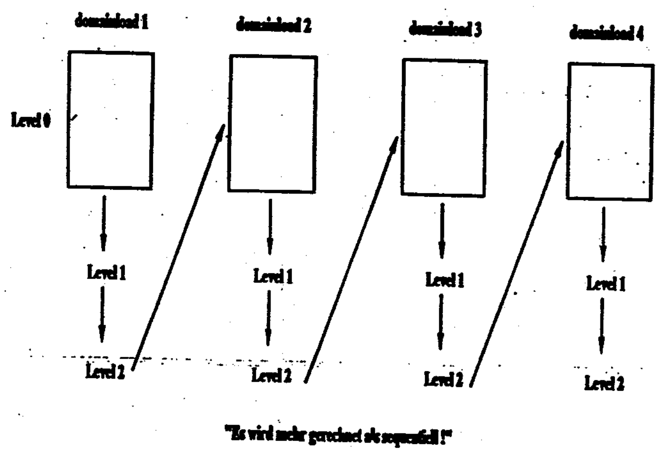


Figure 3: The broadcast approach.



jeweils
alle ranges

Figure 4: Basic pipeline version.

Message → Modeling

COMPRESSOR

Symbol	Observed Probability	Symbol Range
C	0.1	0.0 - 0.1
E	0.1	0.1 - 0.2
M	0.1	0.2 - 0.3
O	0.2	0.3 - 0.5
P	0.1	0.5 - 0.6
R	0.2	0.6 - 0.8
S	0.2	0.8 - 1.0

Coding

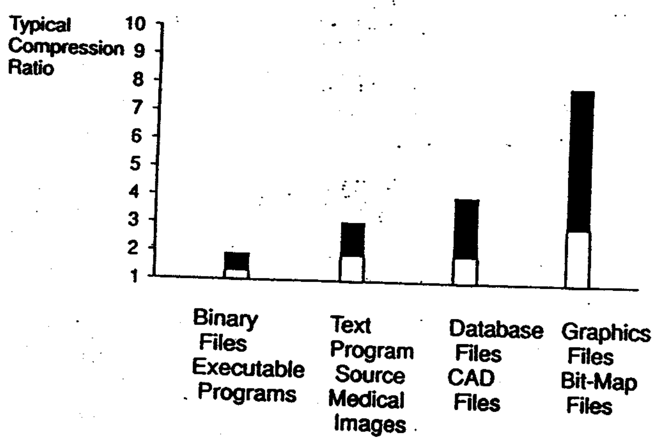
New Symbol	Symbol Range	Interval Width	Message Interval after encoding symbol
			0.0 - 1.0
C	0.0 - 0.1	1.0	0.0 - 0.1
O	0.3 - 0.5	0.1	0.03 - 0.05
M	0.2 - 0.3	0.02	0.034 - 0.036
P	0.5 - 0.6	0.002	0.0350 - 0.0352
R	0.6 - 0.8	0.0002	0.03512 - 0.03516
E	0.1 - 0.2	0.00004	0.035124 - 0.035128
S	0.8 - 1.0	0.000004	0.0351272 - 0.0351280
S	0.8 - 1.0	0.0000008	0.03512784 - 0.03512800
O	0.3 - 0.5	0.00000016	0.035127888 - 0.035127920
R	0.6 - 0.8	0.000000032	0.0351279072 - 0.0351279136

Figure 4.4 – Nonadaptive arithmetic coding example.

Table 4.3 LZW compression example.

Input Symbol	Encoder Dictionary Update	Encoder Output	Decoder Dictionary Update	Decoder Output
D	256 = DA	D		D
A	257 = AT	A	256 = DA	A
T	258 = TA	T	257 = AT	T
A	259 = Aspace	A	258 = TA	A
space	260 = spaceC	sp	259 = Aspace	space
C	261 = CO	C	260 = spaceC	C
O	262 = OM	O	261 = CO	O
M	263 = MP	M	262 = OM	M
P	264 = PR	P	263 = MP	P
R	265 = RE	R	264 = PR	R
E	266 = ES	E	265 = RE	E
S	267 = SS	S	266 = ES	S
S	268 = SI	S	267 = SS	S
I	269 = IO	I	268 = SI	I
O	270 = ON	O	269 = IO	O
N	271 = Nspace	N	270 = ON	N
space	272 = spaceCO	260	271 = Nspace	space
C	—	—	—	C
O	273 = OMP	262	272 = spaceCO	O
M	—	—	—	M
P	274 = PRE	264	273 = OMP	P
R	—	—	—	R
E	275 = ESS	266	274 = PRE	E
S	—	—	—	S
S	276 = SE	S	275 = ESS	S
E	277 = ESspace	266	276 = SE	E
S		—	—	S
space	278 = spaceD	space	277 = ESspace	space
D	279 = DAT	256	278 = spaceD	D
A	—	—	—	A
T	280 = TAspace	258	279 = DAT	T
A	—	—	—	A
space		space	280 = TAspace	space

Abb. X



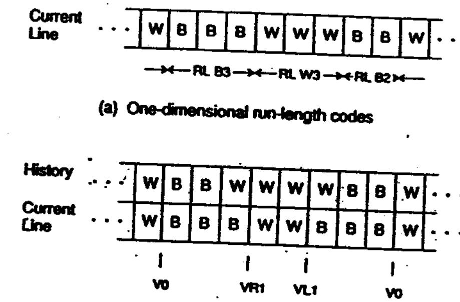
(a) Compression ratio

	Run-Length Coding	Huffman Coding	Arithmetic Coding	LZ77 / LZ78 Dictionary Compression
Compression Ratio	1	3	5	4
Encoding Speed	5	2	1	4
Decoding Speed	5	3	1	4

Low=1 High=5

(b) Algorithm performance

Figure 4.6 Symbolic data compression summary.



(a) One-dimensional run-length codes

(b) Two-dimensional vertical reference codes

Figure 5.7 ITU-T T.4 and T.6 bi-level image coding.

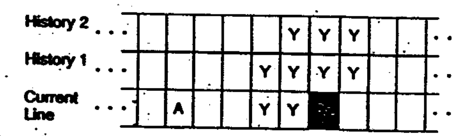


Figure 5.8 JBIG two-dimensional, three-line prediction template.

Test Image	JPEG (H)	JPEG (A)	JPEG-LS	PNG
Natural images				
Barbara	1.48	1.53	1.70	1.53
Lena	1.59	1.59	1.74	1.63
Zelda	1.86	1.90	2.05	1.92
Compound documents				
cmpnd1	3.22	5.62	6.38	5.98
cmpnd2	3.05	5.27	5.86	5.34
Line art				
Einstein	1.74	2.13	2.29	2.07
finger	1.37	1.35	1.41	1.38

Table 2.10 Compression ratios for images with 8-bits per pixel using the JPEG, JPEG-LS, and PNG lossless compression algorithms.

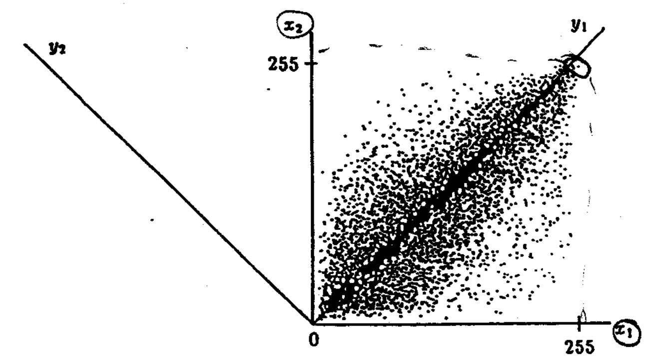


Figure 10.2: Rotation transform example.

Now consider a unitary transform of the vector $\mathbf{X}$ into the vector $\mathbf{Y} = (y_1, y_2)$ by rotating the $x_1 x_2$-coordinate axes by 45^0; i.e.,

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$

or equivalently,

$$\mathbf{Y} = \mathbf{AX}, \tag{10.2}$$

where $\mathbf{A}$ is the rotation matrix. Equation (10.2) defines the *forward transform*. An important feature of a unitary transform is that it is distance preserving; i.e., it does not change the Euclidean distance between vectors.

Since a unitary transform is reversible, the original data can be recovered if no errors are introduced by the encoding process. The rotation transform in the above example can be reversed by performing the *inverse transform*, which is a rotation by -45^0; i.e.,

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix},$$

or in matrix notation,

$$\mathbf{X} = \mathbf{BY}, \tag{10.4}$$

where $\mathbf{B} = \mathbf{A}^{-1}$. It is useful to note that for unitary transforms, $\mathbf{A}^{-1} = \mathbf{A}^{*T}$, where $*$ denotes the complex conjugate. In a practical encoding scheme, the transform coefficients are quantized, and the inverse transform of the quantized coefficients results in an approximation to the original image. Because of the distance-preserving property of the rotation, the MSE between the original image and the reconstructed image is equal to the MSE introduced by the quantization process in the transform domain.

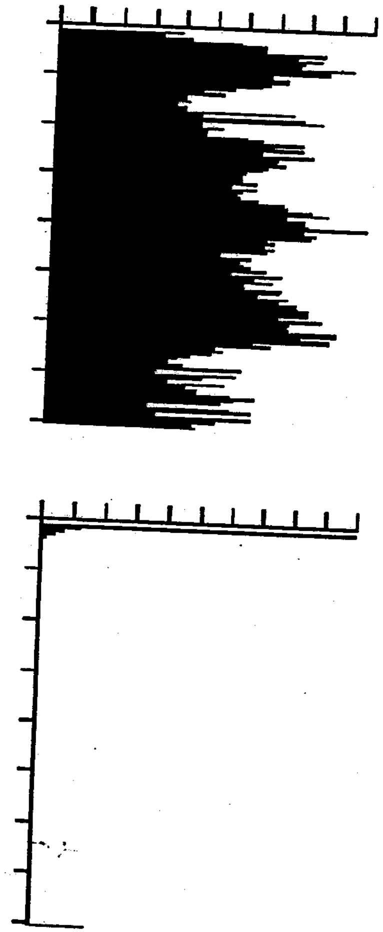


Figure 1: Histogramm of original Lena image and transformed image