

Coding and compression

Michael Lund, ph.d. student



Compression consists of

- An encoding algorithm
 - Takes a message and generates a “compressed” representation.

- A decoding Algorithm
 - Reconstructs the original message or an approximation of it from the compressed representation.



Why use compression

- Compression is useful and necessary in many applications
 - Transmission
 - Storage
- 2 Min. CD-quality uncompressed takes around 84 MB
- A second video without compression takes around 20 MB



How does Compression work?

- Compression basically removes redundancy in the data

Example: 10

Coding: 19 times "10"

- Compression find as short a representation as possible by removing correlation in the data

Temporal - correlation in Audio and video

Spatial - correlation between neighbouring pixels in images and video

Spectral - correlation between colour or luminescence components

"Perceptual - correlation" by exploiting human perceptual properties



-
- Information is measured in number of bits
 - Redundancy of a representation is:

Bits in representation - bits in shortest representation

- Compression ratio is:

Ratio between original file and compressed file



Entropy

- The average information per symbol is called entropy and is defined according to Shannon's formula as

$$H = -\sum_{i=1}^n P_i * \log_2 P_i$$

Where n is the number of different symbols in the source stream and P_i is the probability of occurrence of symbol i.

- This is the theoretical lower bound on the message length
- The actual average information per symbol used is defined as

$$\sum_{i=1}^n N_i * P_i$$

Where N_i is the number of bits used to encode the symbol i.



Entropy of images

- In an image with uniform random distribution of gray-level intensities (8 bit) $p_i = 1/256$

$$H = - 256 * 1/256 * \log_2(1/256) = 8 \text{ bit/char}$$

- In an image with random distribution of pixels where half of the pixels are white ($I = 220$) and the other half are black ($I = 10$), $p_i = 1/2$

$$H = - 2 * \frac{1}{2} * \log_2(\frac{1}{2}) = 1 \text{ bit/char}$$



Entropy of the English Language

- If we assume equal probability for all characters, a separate code for each character and 96 printable characters (standard keyboard

$$H = \log_2(96) = 6.6 \text{ bit/char}$$

- If we assume a probability distribution

$$H = 4.7 \text{ bit/char}$$

- If the text is broken down to blocks of 8 characters

$$H = 2.4 \text{ bit/char}$$



Classification of compression techniques

- Lossless (reversible) compression
 - Decoded signal is identical to original signal.
 - Required for compressing computer programs, legal, and medical documents where no error or loss are allowed.
 - Exploit statistical properties.

- Lossy (irreversible) compression
 - Decoded signal differs from original signal.
 - Used for compressing digital audio, images, and video.
 - Exploit statistical and human perception properties.



Lossless compressions techniques

- All lossless compression techniques relies on uneven distribution of values to be encoded
- In many applications the symbols/code words do not occur with the same frequency of occurrence. Some of the lossless encoding techniques exploit this property by using a set of variable length code words with the shortest code words used to represent the most frequently occurring symbols.
- Lossless compressions techniques
 - Run-Length coding
 - Huffman Coding
 - Lempel-Ziv-Welch Coding
 - Arithmetic coding



Run-Length Coding

- Run Length Encoding (RLE) is a very simple and fast data compression algorithm.
- The repeated occurrence of the same character (a run) is replaced with the repeated character and the length (number of repetition) of the run.
- Useful when source information comprises long substrings of the same character or binary digit.
- An example of a run length encoding:

Uncompressed data: ABCCCCCCCCDEFGGG
Compressed data: AB C!8 DEFGGG

- RLE is often used in connection with other compression techniques
- Used in facsimile (fax) machines



Fixed Length Encoding: American Standard Code for Information Interchange (ASCII)

1	SOH	27	ESC	53	5	79	O	105	i
2	STX	28	FS	54	6	80	P	106	j
3	ETX	29	GS	55	7	81	Q	107	k
4	EOT	30	RS	56	8	82	R	108	l
5	ENQ	31	US	57	9	83	S	109	m
6	ACK	32		58	:	84	T	110	n
7	BEL	33	!	59	;	85	U	111	o
8	BS	34	"	60	<	86	V	112	p
9	TAB	35	#	61	=	87	W	113	q
10	LF	36	\$	62	>	88	X	114	r
11	VT	37	%	63	?	89	Y	115	s
12	FF	38	&	64	@	90	Z	116	t
13	CR	39	'	65	A	91	[117	u
14	SO	40	(66	B	92	\	118	v
15	SI	41)	67	C	93]	119	w
16	DLE	42	*	68	D	94	^	120	x
17	DC1	43	+	69	E	95	_	121	y
18	DC2	44	,	70	F	96	`	122	z
19	DC3	45	-	71	G	97	a	123	{
20	DC4	46	.	72	H	98	b	124	
21	NAK	47	/	73	I	99	c	125	}
22	SYN	48	0	74	J	100	d	126	~
23	ETB	49	1	75	K	101	e	127	DEL
24	CAN	50	2	76	L	102	f		
25	EM	51	3	77	M	103	g		
26	SUB	52	4	78	N	104	h		



Frequency of occurrence

- Given an English text: " Once upon a time"
The characters do not show up equally often.
- "e" is normally more common in English than "q"
- We may select a shorter representation for "e" and a longer for "q"
- The higher the probability, the shorter the code.



Morse Coding

A	.-	K	-.-	U	..-	4-
B	-...	L	.-...	V	...-	5
C	-.-.	M	--	W	.--	6	-....
D	-..	N	-. .	X	-..-	7	--...
E	.	O	---	Y	-. —	8	---..
F	...-	P	.-..	Z	--..	9	----.
G	--.	Q	--.-	0	-----	Full stop	...--
H	R	.-.	1	.----	Comma	--...-
I	..	S	...	2	..----	Query	..--...
J	.----	T	-	3	...--		

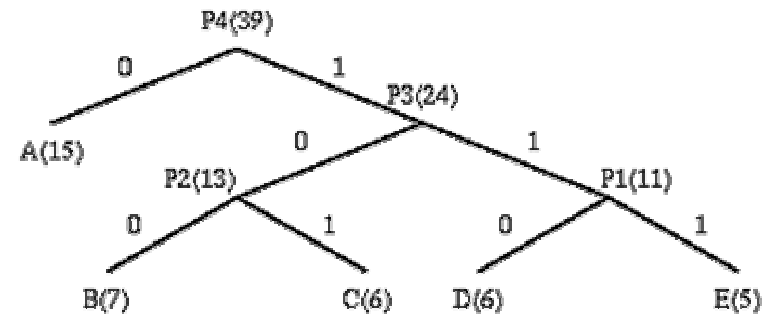


Static Huffman Coding

- Is a prefix-free code and uniquely decodable

String: AABDEAAAEEDDEEBABACCCCACABDDDDAAAAABCBB

Symbol	Count	P	$-P_i \cdot \log P_i$	Code	$P_i \cdot N_i$
A	15	0.385	0.530	0	0.385
B	7	0.179	0.445	100	0.537
C	6	0.154	0.416	101	0.462
D	6	0.154	0.416	110	0.462
E	5	0.128	0.380	111	0.384
	39	1.000	2.187		2.225



Ideally will Huffman coding assign a code of length $-\log_2 P_i$



Text to decode

00100001100100111110011111110000010011101011010011101011

57 bit

d	0000
e	0001
g	011
h	0010
i	0011
l	100
m	0100
n	101
o	111
r	0101
	110



Dynamic Huffman coding

- The previous algorithm require the statistical knowledge which is often not available (e.g.. live audio and video).
- The previous algorithm can result in a large overhead

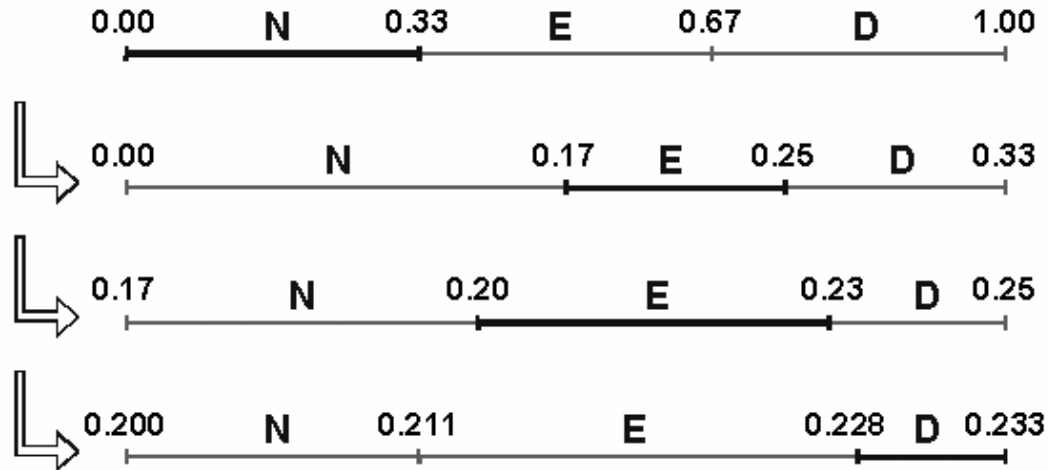


Arithmetic Coding

- The codewords produced always achieve the entropy value

(Huffman coding only achieves the entropy value if all the character/symbol probabilities are integer powers of $\frac{1}{2}$)

Single context example: encoding word **NEEDED**



Lempel-Ziv-Welch Coding

- Encoder and decoder build the content of the dictionary dynamically as the text is transferred.
- Initially the dictionary only holds the character set (e.g. ASCII).
- The remaining entities in the dictionary are then build up dynamically.
- Only strings of alphanumerical characters are stored in the dictionary and other characters are interpreted as word delimiters.
- A 25.000 word dictionary requires 15 bits to encode.



LZW example

Dub I Dub I Dub I Dub Dub Dub Dub I Dub I Dub I, Yeah Yeah!



Entropy Encoding Summary

- Huffman maps fixed length symbols to variable length codes. Optimal only when symbol probabilities are powers of $\frac{1}{2}$.
- Arithmetic maps entire message to real number range based on statistics. Theoretically optimal for long messages, but optimality depends on data model. Also can be CPU/memory intensive.
- Lempel-Ziv-Welch is a dictionary-based compression method. It maps a variable number of symbols to a fixed length code.
- Adaptive algorithms do not need a priori estimation of probabilities, they are more useful in real (time) applications.



Predictive/differential coding

- The difference between a prediction value and actual sample value is coded and transmitted. Differential pulse-coded modulation (DPCM)
- The predicted error will normally be smaller than the original sample value. (save typical 1 bit)
- If the predicted error is larger than the quantization steps, the original audio quality will not be maintained.
- With Adaptive DPCM an eight-order predictor is used and the quantization step varied according to the signal itself.

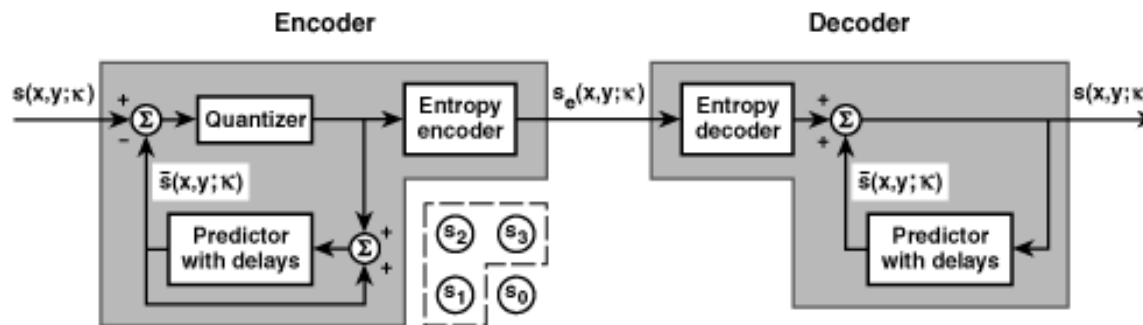


Figure. Differential pulse code modulation (DPCM) with entropy coding.

ADPCM Standards

	Compression technique	Speech bandwidth (kHz)	Sampling rate(kHz)	Compressed bit rate (kbps)
G.711	PCM (no compression)	3.4	8	64
G.721	ADPCM	3.4	8	32
G.722	Subband ADPCM	7	16 (14 bits)	48,56,64
G.723	ADPCM	3.4	8	24



Linear predictive coding

- Quantification of perceptual features.
- Can sound synthetic, but very high levels of compression can be achieved.
- Important parameters
 - Pitch (the ear is more sensitive to frequencies in the range 2-5kHz)
 - Vocal tract excitation parameters
 - Voiced (m,v,l) and unvoiced sounds (f,s)
- With a suitable model of the vocal tract is it possible to generate a synthesized version of the original speech signal.

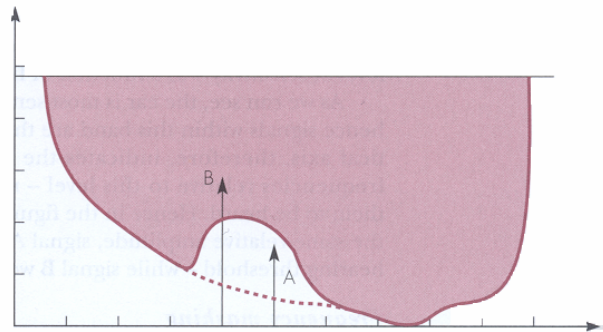
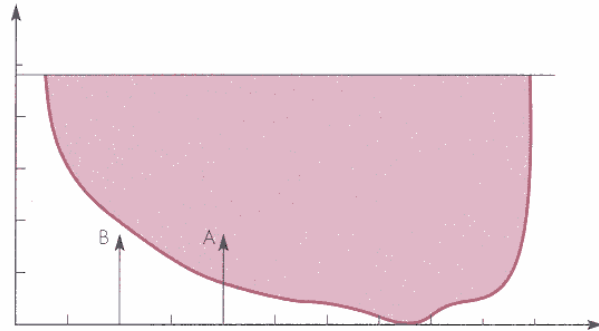


Perceptual coding

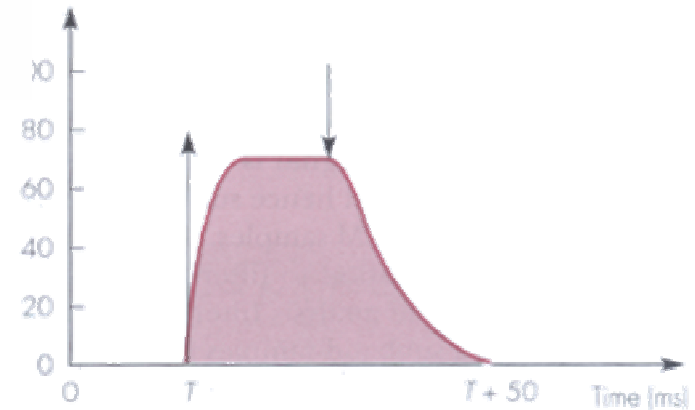
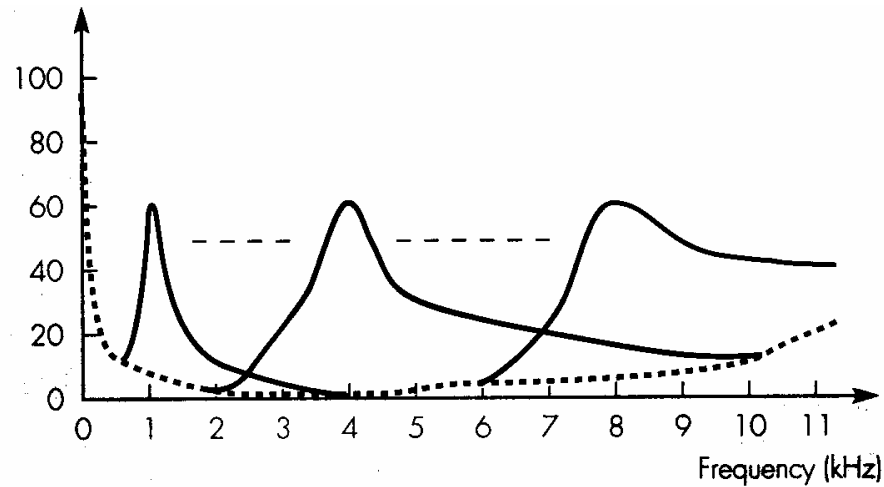
- A sound can alter the appearance of another sound
- Perceptual coding uses psychoacoustic models based on the limitations of the ears.
- Frequency masking
- Temporal masking
- Source audio waveform analyzed - only features that are perceptible to the ear are transmitted.



Frequency masking



Frequency and Temporal masking

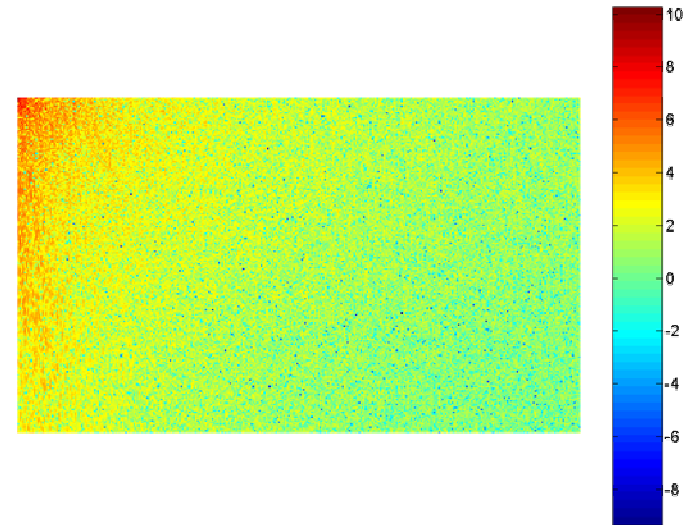


Transformation encoding

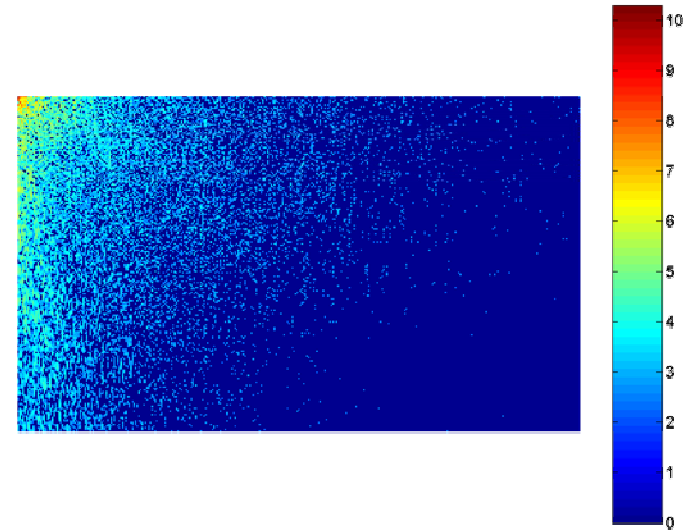
- Transforming the source information from one form into another (easier to compress).
- There is no data loss involved in the transform process.
- Discrete cosine transform (DCT)
- Discrete Fourier transform (DFT)
- The human eye is less sensitive to higher spatial frequency components than the lower.
-



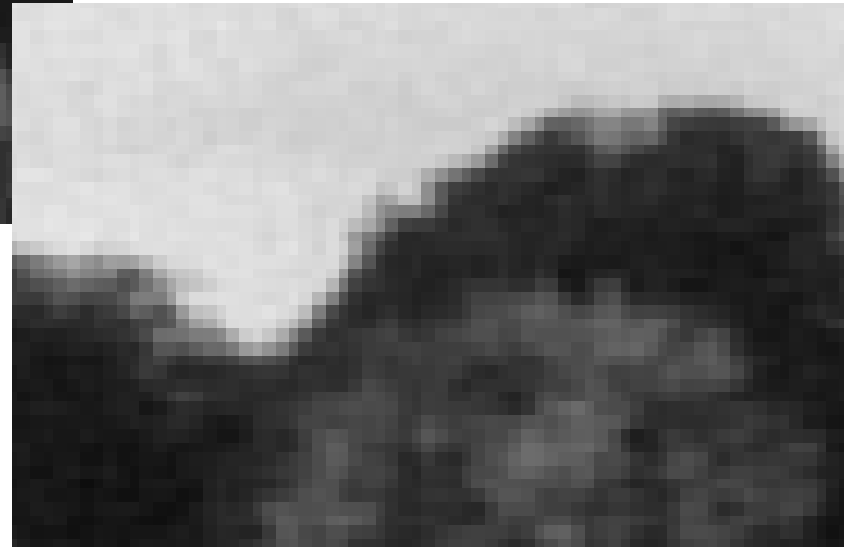
DCT transform



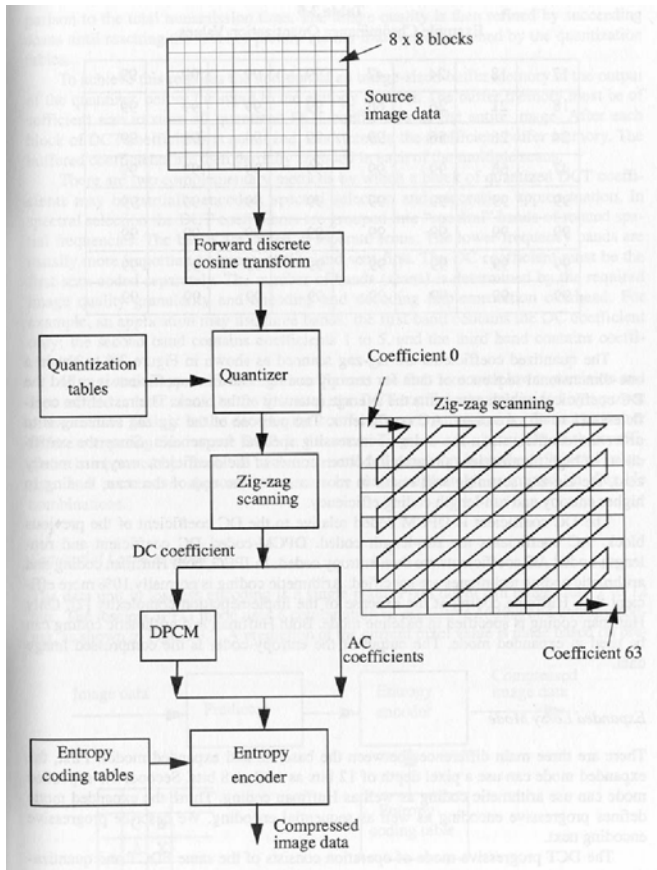
DCT transform thresholded



DCT transform closeup



JPEG



Steps Involved:

- Color transformation RGB to YIQ (optional, gives better compression)
- Split image in 8x8 blocks
- Discrete Cosine Transform of each block
- Quantization using a table or using a constant (reduce number of bits per sample)
- Zig-Zag scan (group frequencies)
- Differential Pulse Code Modulation (DPCM) on the DC component and Run length Coding of the AC components
- Entropy coding (Huffman) of the final output

Compression performance

- Compression ratio
- Reconstructed media quality
- Implementation complexity
- Compression speed



