

Tipi di Informazione

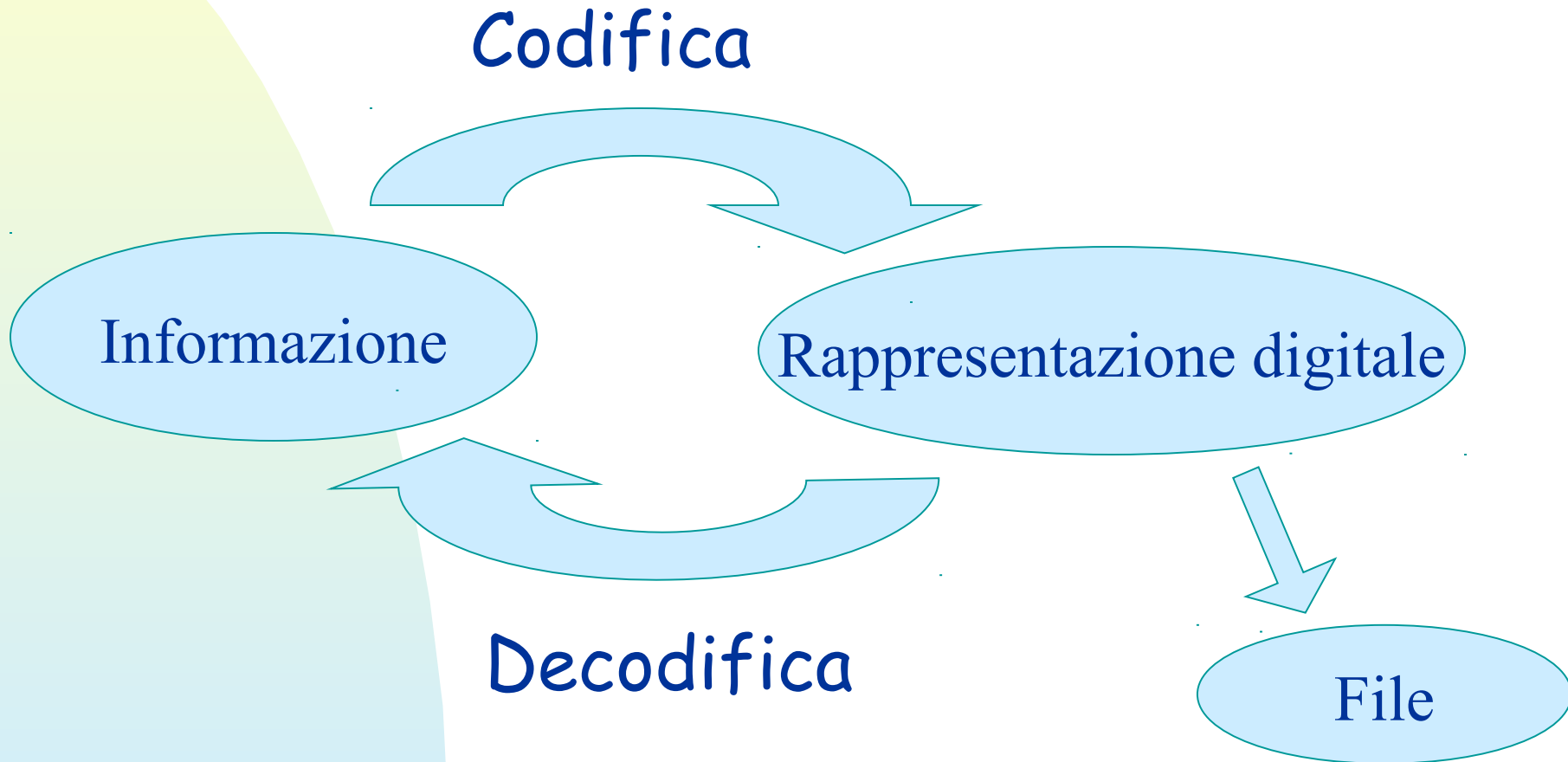
- Testo (caratteri)
- Numeri
- Suoni
- Immagini

Le informazioni sono concetti astratti che esistono indipendentemente dalla loro rappresentazione

Tutto ciò che ci circonda è informazione



Informazione digitale



Codifica dell'informazione

Idea di base: Usare

- presenza/assenza di carica elettrica
- passaggio/non passaggio di corrente/luce

BInary digi**T** (cifra binaria): il BIT

Usiamo cioè una rappresentazione binaria (a due valori) dell'informazione



Codifica dell'informazione

Con 1 bit rappresentiamo solo 2 diverse informazioni:

si/no - on/off - 0/1 - vero/falso

Mettendo insieme piu' bit possiamo rappresentare piu' informazioni:

2 bit = 4 informazioni

00 - 01 - 10 - 11



Codifica dell'informazione

- **Esempio:** un esame può avere quattro possibili esiti
 - ottimo 00
 - discreto 01
 - sufficiente 10
 - insufficiente 11

- Con 2 bit si codificano 4 informazioni (2^2)
- Con 3 bit si codificano 8 informazioni (2^3)
-
- Con N bit si codificano 2^N informazioni



Codifica dell'informazione

Se dobbiamo rappresentare più di 4 informazioni occorre aggiungere il 3° bit, tante parole quanti sono i concetti

000, 001,

010, 011,

100, 101

110, 111



Codifica dell'informazione

Se dobbiamo rappresentare 21 concetti diversi mi servono

$$2^N \geq M \text{ dove } M=21$$

$$2^5 = 32$$

Alcune sequenze (da 22 a 32) non vengono utilizzate



Codifica dell'informazione

Per rappresentare **57** informazioni diverse dobbiamo usare gruppi di almeno **6** bit.

Infatti:

$$2^6 = 64 > 57$$

Cioe' un gruppo di 6 bit puo' assumere 64 configurazioni diverse:

000000 / 000001 / 000010 .../ 111110 / 111111



Codifica dell'informazione

In generale, con N bit, ognuno dei quali può assumere 2 valori, possiamo rappresentare 2^N informazioni diverse

viceversa:

Per rappresentare M informazioni dobbiamo usare N bit, in modo che: $2^N \geq M$



Il Byte

In informatica ha assunto particolare importanza il concetto di:

$$\text{byte} = 8 \text{ bit} = 2^8 = 256 \text{ inf. diverse}$$

Il byte e' usato come unita' di misura per indicare le dimensioni della memoria, le dimensioni del disco, la potenza di un elaboratore,

Usando sequenze di byte (e quindi aggregati di 8 bit di bit) si possono rappresentare caratteri, numeri immagini, suoni.



Multipli del Byte

Di solito si usano i multipli del byte

Kilo KB 2^{10} (~ mille byte, 1024)

MegaMB 2^{20} (~ un milione , 1KBx1024)

Giga GB 2^{30} (~ un miliardo, 1MBx1024)

Tera TB 2^{40} (~ mille miliardi, 1GBx1024)



Codifica dei caratteri

E' necessario individuare un codice numerico per rappresentare i caratteri.

Il codice **ASCII** (American Standard Code for Interchange Code)

usa i primi 7 bit di ogni byte:

$2^7 = 128$ caratteri diversi

Sufficienti per l'alfabeto anglosassone



Codifica dei caratteri

Per la codifica di 127 caratteri standard sono sufficienti 7 bit (ASCII standard) - Talvolta occorre utilizzare

- 8 bit (ASCII esteso) 256 caratteri

Oppure ancora

- 16 bit (UNICODE) 65535 caratteri
- MS Windows usa un codice proprietario a 16 bit per carattere, simile ad UNICODE ma non totalmente standard chiamato multilanguage
- 8 bit (EBCDIC) usato per i display lcd



Codifica ascii dei caratteri

ASCII	Simb.	ASCII	Simb.	ASCII	Simb.
00000000	NUL	00001110	SO	00011100	FS
00000001	SOH	00001111	SI	00011101	GS
00000010	STX	00010000	DLE	00011110	RS
00000011	ETX	00010001	DC1	00011111	US
00000100	EOT	00010010	DC2	00100000	SP
00000101	ENQ	00010011	DC3	00100001	!
00000110	ACK	00010011	DC4	00100010	"
00000111	BEL	00010101	NAK	00100011	#
00001000	BS	00010110	SYN	00100100	\$
00001001	HT	00010111	ETB	00100101	%
00001010	NL	00011000	CAN	00100110	&
00001011	VT	00011001	EM	00100111	'
00001100	NP	00011010	SUB	00101000	(
00001101	CR	00011011	ESC	00101001)

Codifica ascii dei caratteri

ASCII	Simb.	ASCII	Simb.	ASCII	Simb.
00101010	*	00111001	9	01000111	G
00101011	+	00111010	:	01001000	H
00101100	,	00111011	;	01001001	I
00101101	-	00111100	<	01001010	J
00101110	.	00111101	=	01001011	K
00101111	/	00111110	>	01001100	L
00110000	0	00111111	?	01001101	M
00110001	1	01000000	@	01001110	N
00110010	2	01000001	A	01001111	O
00110011	3	01000010	B	01010000	P



Codifica ascii dei caratteri

ASCII	Simb.	ASCII	Simb.	ASCII	Simb.
00110100	4	01000011	C	01010001	Q
00110101	5	01000100	D	01010010	R
00110110	6	01000101	E	01010011	S
00111000	8	01000110	F	01010100	T



Codifica delle parole

Sono sequenze di caratteri

- Esempio: **INFORMATICA**

01001001 01001110 01000110 01001111 01010010 01001110 01000001
I N F O R M A

01010100 01001001 01000011 01000001
T I C A



Codifica dei numeri (decimale)

Sistema posizionale in cui ogni cifra di un numero assume un valore che dipende dalla sua posizione

Il sistema decimale (base 10):

$$245 = 2 \times 100 + 4 \times 10 + 5 \times 1$$

$$851 = 8 \times 10^2 + 5 \times 10^1 + 1 \times 10^0$$

*Si deve fare la somma dei prodotti di ciascuna cifra moltiplicata per la base elevata all'esponente che rappresenta la **posizione** della cifra stessa (partendo da 0)*



Codifica dei numeri (binario)

Utilizza una notazione posizionale basata su 2 cifre (0 e 1) e sulle potenze di 2

Esempio: 10110 =

$$1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 22$$

Esempio: 1010101 =

$$1 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 85$$

$$1 \times 64 + 0 \times 32 + 1 \times 16 + 0 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1 = 85$$

Conversioni dei numeri (da $_{10}$ a $_2$)

Per convertire un numero in base 2 si devono trovare i resti delle divisioni successive del numero per la base 2

Esempio: 157_{10}

$157 / 2$ resto 1

$78 / 2$ resto 0

$39 / 2$ resto 1

$19 / 2$ resto 1 10011101

$9 / 2$ resto 1

$4 / 2$ resto 0

$2 / 2$ resto 0

$1 / 2$ resto 1



Conversioni dei numeri (da $_{10}$ a $_2$)

Un metodo pratico

Esempio: 157_{10}

128	64	32	16	8	4	2	1
1	0	0	1	1	1	0	1
29		13	5	1		0	

Conversioni dei numeri (da $_2$ a $_{10}$)

Un metodo pratico

Esempio: 10011101

128	64	32	16	8	4	2	1
1	0	0	1	1	1	0	1
128+			16 + 8 + 4 +			1	= 157

Conversioni dei numeri (da $_2$ a $_{16}$)

Un metodo pratico

Esempio: 10011101

8	4	2	1		8	4	2	1
1	0	0	1		1	1	0	1
	9				D			= 157

Addizione binaria

$$\begin{array}{r} 10111001 + (185) \\ 10000100 = (132) \\ ===== \\ 10011101 \quad (317) \end{array}$$

Overflow = oltre in numero massimo di cifre rappresentabili con un particolare numero di cifre (es Byte)

Sottrazione binaria

$$\begin{array}{r} 10111001 - \quad (185) \\ \quad 10100 = \quad (20) \\ \hline 10100101 \quad (165) \end{array}$$



Esercizi

■	111011	59
■	101111101	381
■	110000	48
■	110010101	405
■	101010101010	2730
■	101000000001	2561



Esercizi

128

10000000

1234

10011010010

5654

1011000010110

1233

10011010001

234

11101010

511

111111111



Cifre rappresentabili

Sist. Decimale = 99...99 = $10^N - 1$

Sist. Binario = 11...11 = $2^N - 1$

Esempio: 11111111 (8 bit) = $2^8 - 1 = 255$.

Per rappresentare il n. 256 serve un bit in

piu': $100000000 = 1 * 2^8 = 256$.

Cifre rappresentabili

Fissate quante cifre (bit) sono usate per rappresentare i numeri, si fissa anche il numero piu' grande che si puo' rappresentare

- ◆ con 16 bit: $2^{16} - 1 = 65.535$
- ◆ con 32 bit: $2^{32} - 1 = 4.294.967.295$
- ◆ con 64 bit: $2^{64} - 1 = \text{circa } 1,84 * 10^{19}$

Si possono rappresentare numeri piu' grandi se si tollera un certo grado di imprecisione.

Overflow

Deciso il numero di cifre a disposizione si fissa anche il **numero massimo** rappresentabile, numeri più grandi causano problemi di **overflow**

Esempio: 4 cifre

in base 10 $9999 + 1 = 10000_{10}$

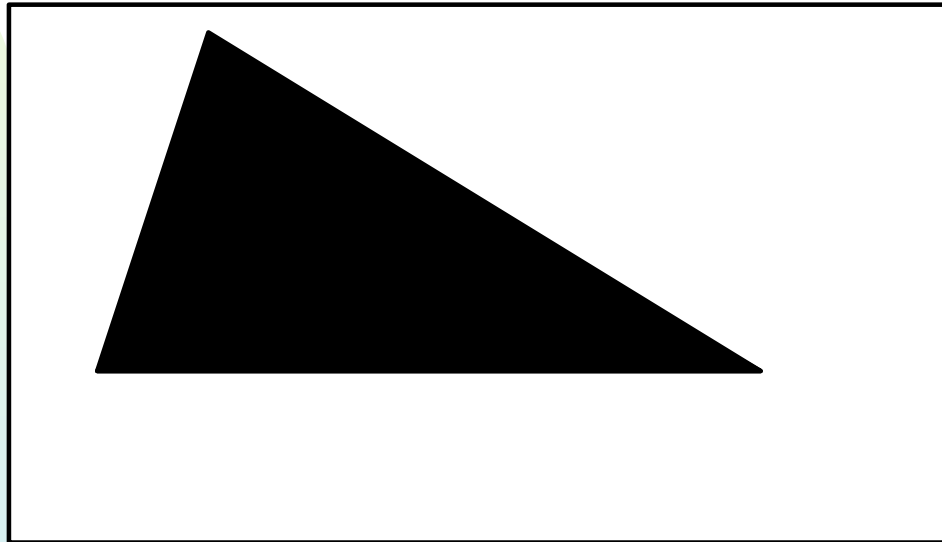
in base 2 $1111 + 1 = 10000_2 (= 16_{10})$

in base 16 $FFFF + 1 = 10000_{16} (= 65536_{10})$

in base 8 $7777 + 1 = 10000_8 (= 4096_{10})$

Rappresentazione delle immagini

Immagini in bianco e nero



Rappresentazione delle immagini

Dividere l'immagine in una griglia a righe orizzontali e verticali

Ogni quadratino della griglia e' un **pixel**

Codificare ogni pixel con:

- 0 se il pixel e' bianco

- 1 se il pixel e' nero

Il formato dell'immagine (BMP, Jpeg, Tiff, ecc.)

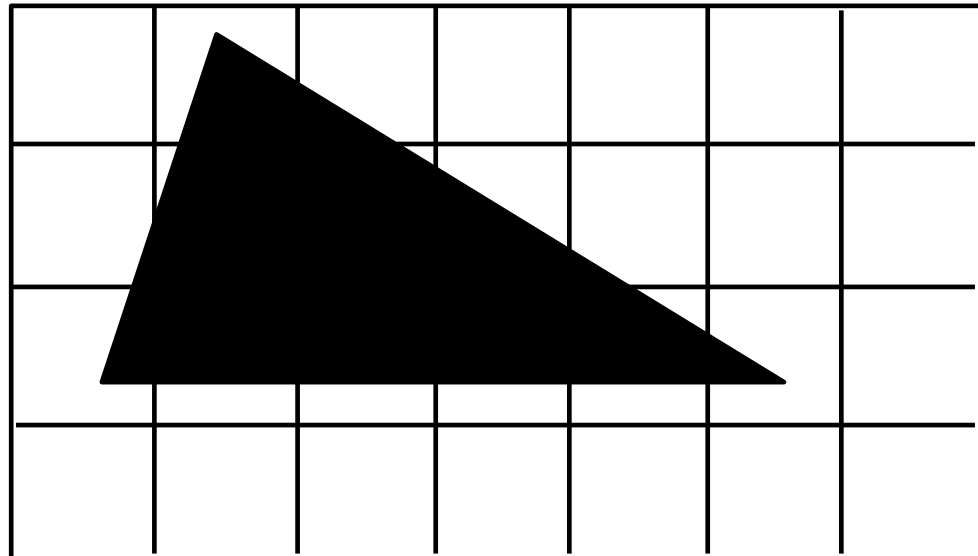
determina **anche** la partenza:

convenzionalmente si parte dal 1° quadratino basso a sinistra.



Rappresentazione delle immagini

Risoluzione 7 X 4



Rappresentazione delle immagini

0000000 0111100 0110000 0100000

0 ₂₂	1 ₂₃	0 ₂₄	0 ₂₅	0 ₂₆	0 ₂₇	0 ₂₈
0 ₁₅	1 ₁₆	1 ₁₇	0 ₁₈	0 ₁₉	0 ₂₀	0 ₂₁
0 ₈	1 ₉	1 ₁₀	1 ₁₁	1 ₁₂	0 ₁₃	0 ₁₄
0 ₁	0 ₂	0 ₃	0 ₄	0 ₅	0 ₆	0 ₇

Risoluzione in bit B/W

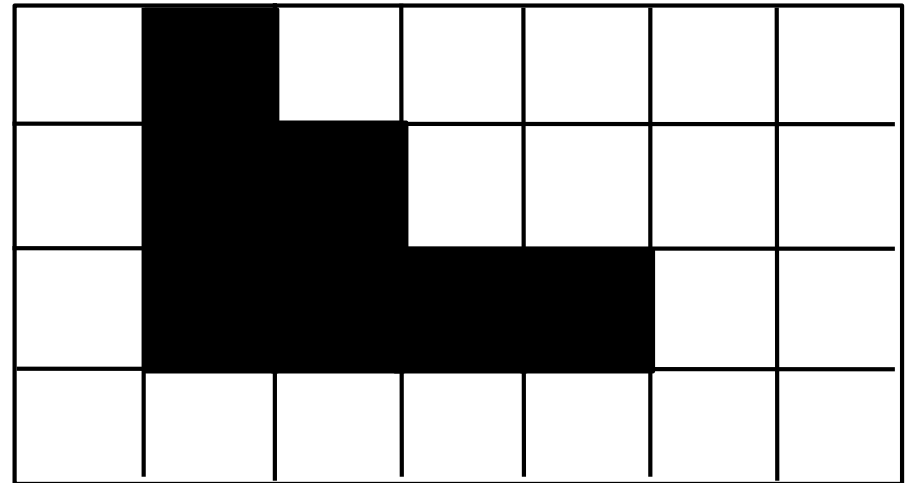
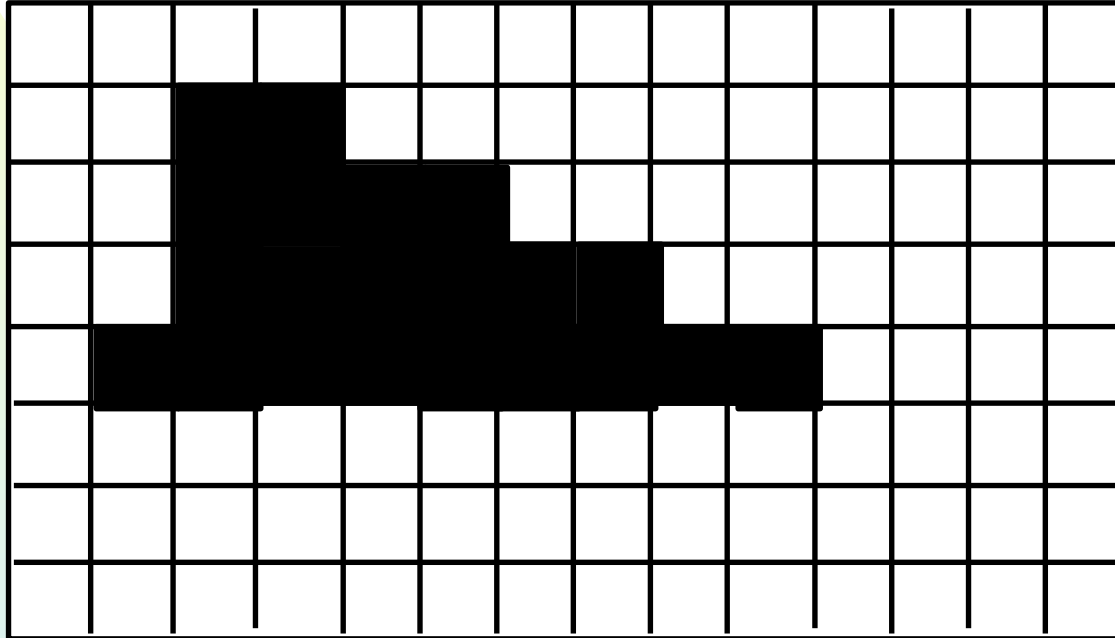


Immagine ricostruita

Rappresentazione delle immagini



Aumentiamo la risoluzione a 14 X 8

Rappresentazione delle immagini

Quindi: le immagini sono rappresentate con un certo livello di approssimazione, o meglio, di **risoluzione**, ossia il numero di pixel usati per riprodurre l'immagine:

- 640 x 480 pixel; 800 x 600 pixel
- 1024 x 768 pixel; 1280 x 1024 pixel
- 1400 x 1280 pixel

É importante anche il **dot pitch**, il grado di definizione del pixel: 0,25 - 0,28 in quanto maggior risoluzione e minor dot-pitch garantiscono immagini migliori

Rappresentazione delle immagini

- Esistono immagini in B/W (1 solo bit per il colore)
- Oppure immagini a livelli di grigio o colori
- Con 8 bit si codificano 256 livelli di grigio. (ottenuti regolando la luminosita' del pixel)
- Con 8 bit si rappresentano 256 colori, con 16 bit 64.000 colori, con 24 bit 16 milioni di colori diversi per pixel e con 32 bit sempre 16 milioni di colori in true color

- **Attenzione** : fare sempre attenzione nel calcolo
a bit e byte 256 colori = 1 byte per pixel



Rappresentazione delle immagini

1 pixel a	2 colori	1 bit
1 pixel a	256 colori	1 byte (1 * 8 bit)
1 pixel a	65535 colori	2 byte (2 * 8 bit)
1 pixel a	16 Mil. di colori	3 byte (3 * 8 bit)



La compressione delle immagini

- **Compressione senza perdita di dati (lossless compression).**
Aree contenenti pixel dello stesso colore vengono codificate in modo compatto; questo permette una compressione limitata ma salvaguarda dalla perdita di informazioni nella fase di decompressione.
- **Compressione con perdita di dati (lossy compression).**
Questa famiglia di tecniche di compressione permette una riduzione nelle dimensioni dell'immagine compressa fino anche a dieci volte rispetto allo schema precedente ma comporta una perdita di informazioni.



La compressione delle immagini

BITMAP (raw): la rappresentazione di una immagine con codifica dei pixel.

Porta via molto spazio:

(immagine 640 x 480 x 32 = Byte)

BITMAP (RLE) - compressa (lossless)

RLE (Run Length Encoding)

Esistono vari formati di codifica:

(giff, jpg, pcx, tiff, ecc.)

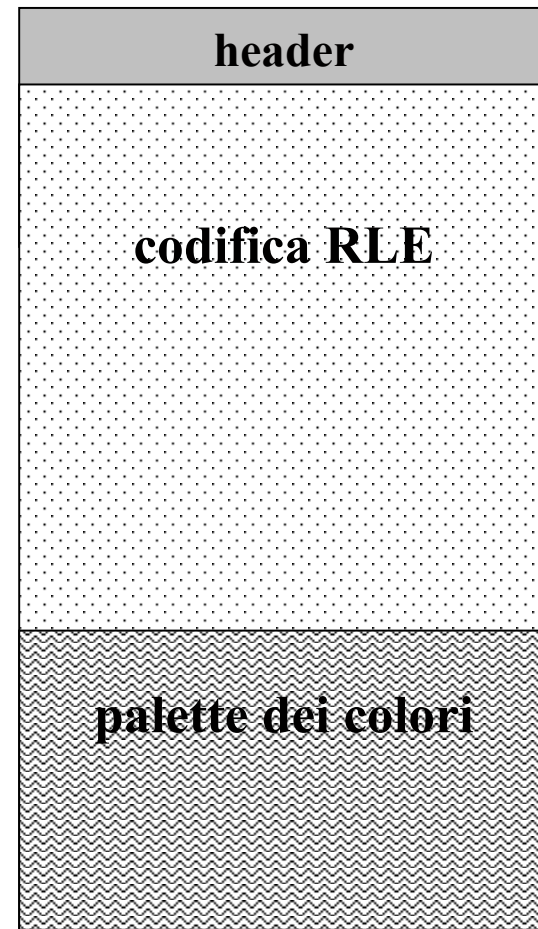
con diversi fattori di compressione in base alla tecnica utilizzata.



La compressione delle immagini

La struttura di un file PCX

lossless
con riduzione di spazio
del 30-40%



La compressione delle immagini

Altri protocolli di conversione

GIF (Graphics Interchange Format), GIF87a o GIF89a

Usa l'algoritmo LZW - tipico di Internet, immagini *interlacciate*, sequenze di caricamento ma limitato a 256 colori.

JPEG (Joint Photographic Experts Group)

JPEG–LS (lossless)

PNG (Portable Network Graphics) lossy

TIFF è strutturato a blocchi chiamati tag



Le immagini vettoriali

- Se le immagini sono regolari si può usare una codifica di tipo **vettoriale** in cui non si specificano le informazioni di colore dei singoli pixel ma ogni elemento geometrico primitivo viene specificato individualmente
- Le immagini vengono costruite a partire dalla descrizione degli elementi che le compongono mediante un linguaggio testuale
- Spesso occupano meno spazio rispetto alle immagini bitmap

La codifica dei filmati

Sono sequenze di immagini **compresse**: (ad esempio si possono registrare solo le variazioni tra un fotogramma e l'altro)

Utilizzano tutti la tecnica del CODEC (COmpress od DECompress)

- Esistono vari formati per codificare filmato e sonoro :

- mpeg (il piu' usato)
- avi (microsoft)
- quicktime (apple) - mov

La codifica dei filmati

CODEC (COmpress od DECompress)

Programmi per la codifica - decodifica flusso dati

MPEG - (Moving Picture Experts Group) fino a 200:1

DIVX

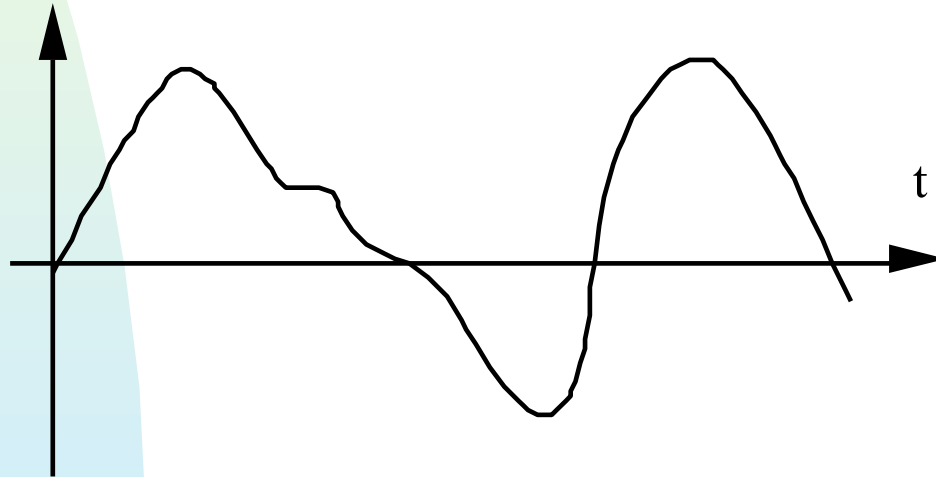
MOTION JPEG (Max 100:1)



Il suono

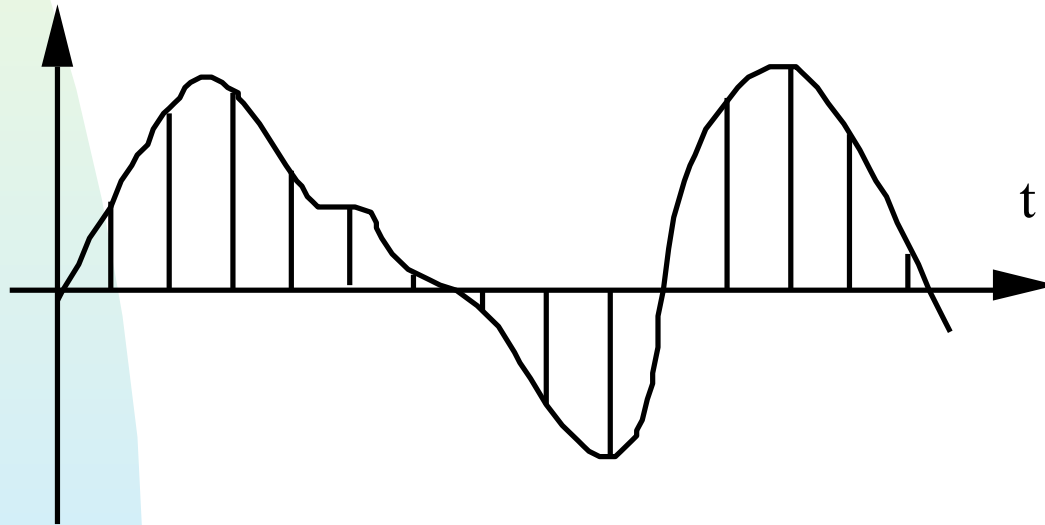
Fisicamente un suono è rappresentato come un'onda che descrive la variazione della pressione dell'aria nel tempo (onda sonora)

Sull'asse delle ascisse viene posto il tempo t e sull'asse delle ordinate la variazione della pressione corrispondente.



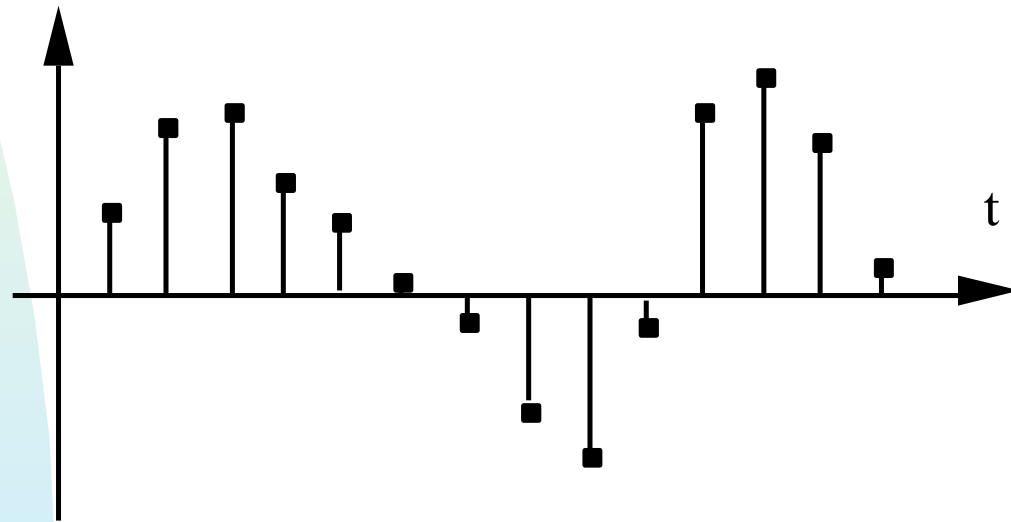
Il campionamento dei suoni

- Si effettuano dei campionamenti sull'onda (cioè si misura il valore dell'onda ad intervalli di tempo costanti) e si codificano in forma digitale le informazioni estratte da tali campionamenti.
- La sequenza dei valori numerici ottenuta dai campioni può essere facilmente codificata



Il campionamento dei suoni

- Quanto più frequentemente il valore dell'onda viene campionato, tanto più precisa sarà la sua rappresentazione
- Il numero di campioni raccolti per ogni secondo definisce la frequenza di campionamento che si misura in Hertz (Hz)



Il campionamento dei suoni

L'onda sonora viene misurata (campionata) ad intervalli regolari

Minore e l'intervallo di campionamento e maggiore e la qualità del suono

La risoluzione standard è a 16 bit

Es. I CD musicali: 44000 campionamenti al secondo, 16 bit per campione.



La codifica dei suoni

Alcuni formati:

- .aiff
- .wav (wave)
- .mpeg
- .cda
- .MP3
- midi usato per l'elaborazione della musica al computer

Esercizi 1

-Abbiamo un'immagine 10 X 20 pixel con 200 colori

-Quanti BIT occupa ?

-Quanti bit occuperebbe se raddoppiassimo i colori ?

-Quanti bit occuperebbe se raddoppiassimo i pixel ?



Esercizi 1s

Per 200 colori occorrono 8 bit (256)

Quindi $10 \times 20 = 200$ pixel

- Per ogni pixel 8 bit, quindi $200 \times 8 = 1600$ bit

- Se raddoppio i colori (da 200 a 400) occorrono 9 bit (512)

- Quindi $200 \times 9 = 1800$ bit (quindi **non** raddoppia lo spazio occupato)

- Se raddoppio i pixel (da 200 a 400)

allora $400 \times 8 = 3200$ bit (raddoppia lo spazio)



Esercizi 2

-10 secondi di suono campionati a 16 bit occupano 2000 byte

-Qual è la frequenza di campionamento ?

-E se raddoppia la frequenza qual è occupazione in bit ?



Esercizi 2s

2000 byte = 16000 bit (2000*8) in 10 secondi

In un secondo 1600 bit (16000/10)

- Se usiamo 16 bit per campione la frequenza è di **100 Hz**

-Se la frequenza diventa 200 Hz allora

$$200 * 16 = 3200$$

$$3200 * 10 = 32000 / 8 = \mathbf{4000 \text{ byte}}$$



Esercizi 3

Un'immagine $20 * 30$ pixel occupa 2400 bit.

- Quanti i colori possibili ?



Esercizi 3 s

Totale pixel $20 * 30 = 600$ pixel

2400 bit diviso i 600 pixel = 4

Quindi 4 bit per pixel che contengono max **16 colori**



Esercizi 4

Abbiamo 5 secondi di suono campionati ad una frequenza di 20 Hz. Con campioni da 2 Byte.

- Quale occupazione in bit
- Se i secondi raddoppiano e la frequenza si dimezza quale occupazione ?

Esercizi 4s

- 5 secondi * 20 Hz = 100 campioni

100 * 16 = 1600 bit

Oppure

16 * 5 = 80 e poi 80 * 20 = 1600 il risultato è il medesimo.

- 10 secondi * 10 Hz = 100 campioni - Non cambia il risultato finale