



**Universita' di Verona  
Dipartimento di Informatica**



# **Compressione dei dati multimediali**

**Davide Quaglia  
a.a. 2009/2010**

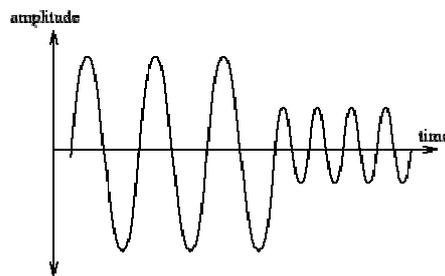
1

## **Natura dei dati multimediali**

2

## Generazione dei suoni

- I **suoni** sono generati da onde di pressione dell'aria che fanno vibrare la membrana dell'orecchio
  - L'**ampiezza delle vibrazioni** determina l'intensità (volume) del suono.
  - La **frequenza di vibrazione** determina l'acutezza del suono.
  - Il segnale ha sempre **media nulla**



3

## Generazione delle immagini

- La luce colpisce la retina e genera un segnale nervoso.
- La lunghezza d'onda della luce determina i colori.
- Esiste una distanza minima sotto la quale due punti sono percepiti dall'occhio come un unico punto (**risoluzione spaziale**).
- Esiste un intervallo di tempo minimo sotto il quale due immagini consecutive danno la percezione del movimento (**risoluzione temporale**).

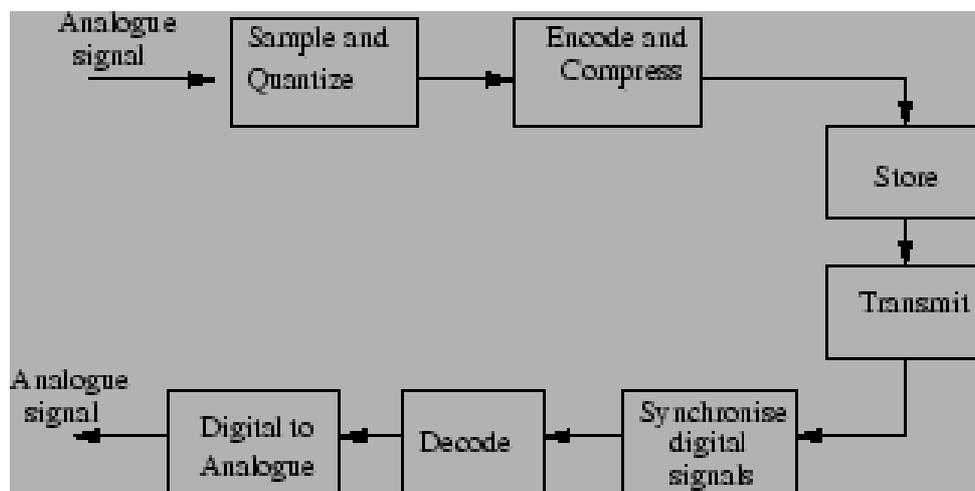
4

## Conversione analogico/digitale

- La pressione dell'aria o l'intensità di luce sono segnali che variano nel tempo assumendo **valori continui** esprimibili mediante **numeri reali** (**segnale analogico**).
- Affinché tali valori possano essere elaborati e compressi occorre trasformarli in una **sequenza discreta** di numeri binari interi con **precisione finita** (**segnale digitale**)
- Due azioni:
  - campionamento
  - quantizzazione

5

## Conversione analogico/digitale



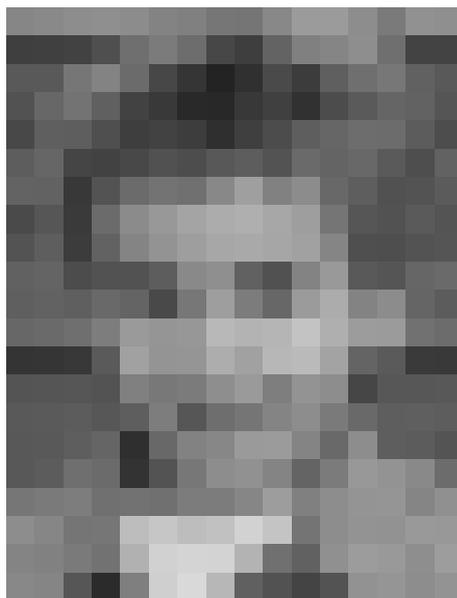
6

## Campionamento di un segnale

- Per catturare la variazione di un segnale nel tempo (suono) o nello spazio (immagine) occorre ripetere la **misura** ad intervalli regolari di tempo o di spazio.
- Intuitivamente più il segnale varia velocemente (nel tempo o nello spazio) e maggiore deve essere la **frequenza di campionamento**.

7

## Quanti campioni ?



16 x 21 pixel

8

## Quanti campioni ?



64 x 85 pixel

9

## Quanti campioni ?



128 x 171 pixel

10

## Quanti campioni ?

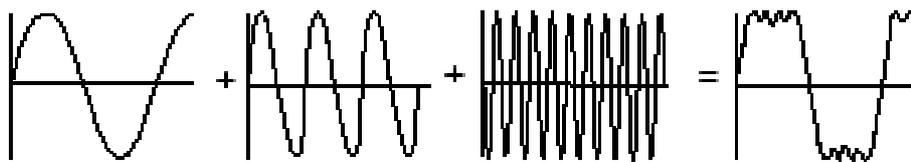


512 x 683 pixel

11

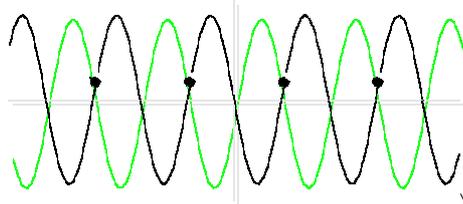
## Teorema del campionamento

- Un segnale può sempre essere rappresentato come **somma di sinusoidi** (rappresentazione in frequenza).
- Per ricreare fedelmente un segnale occorre catturare **tutte** le sinusoidi che lo compongono.



12

## Teorema del campionamento (2)

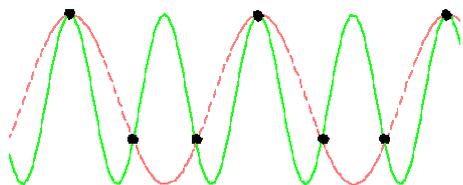


1 campione per periodo

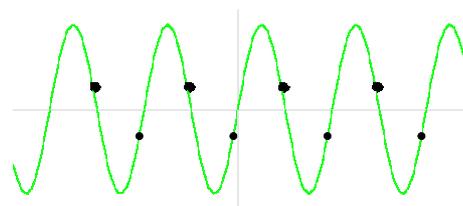
aliasing

13

## Teorema del campionamento (3)



1.5 campioni per periodo



2 campioni per periodo

14

## Teorema del campionamento (4)

- Per riprodurre fedelmente una sinusoide occorrono 2 campioni per periodo.
- La frequenza di campionamento deve essere doppia della frequenza della sinusoide.

15

## Teorema del campionamento (5)

- Se il segnale da campionare ha **larghezza di banda  $f$**  allora la sua componente sinusoidale a frequenza più alta ha periodo  $T=1/f$ .
- Quindi la frequenza di campionamento di tale segnale deve essere doppia della larghezza di banda del segnale (**Teorema di Nyquist**).

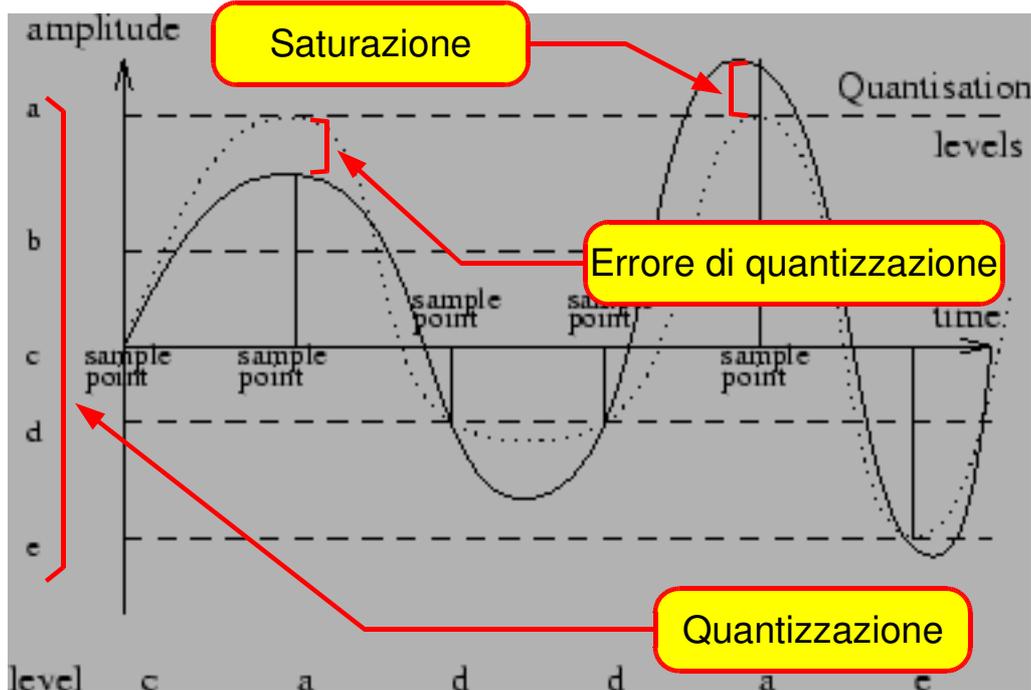
16

## Quantizzazione di una misura

- Assegnazione di una misura a precisione infinita ad un numero avente precisione finita.
  - 1) Determinazione di un range di variazione.
  - 2) Individuazione di un insieme di punti all'interno di tale range.
  - 3) Assegnamento di un numero intero a ciascun punto.
  - 4) Misurazione del segnale analogico e rappresentazione della misura con il numero corrispondente al punto più vicino.

17

## Quantizzazione: problemi



18

## Effetto della quantizzazione sulle immagini



256 livelli (8 bit)



8 livelli (3 bit)

19

## Effetto della quantizzazione

- La quantizzazione implica sempre un errore di arrotondamento detto di **errore di quantizzazione**.
- Maggiore è il numero di bit su cui si rappresenta la misura e minore è l'errore di quantizzazione.
- Tale errore è irreversibile.
  - Quando un procedimento di compressione prevede uno o più fasi di quantizzazione allora non è possibile ritornare alla rappresentazione originale.
- **Compressione con perdita (lossy)**
- La codifica di dati multimediali comporta sempre una perdita di informazione.

20

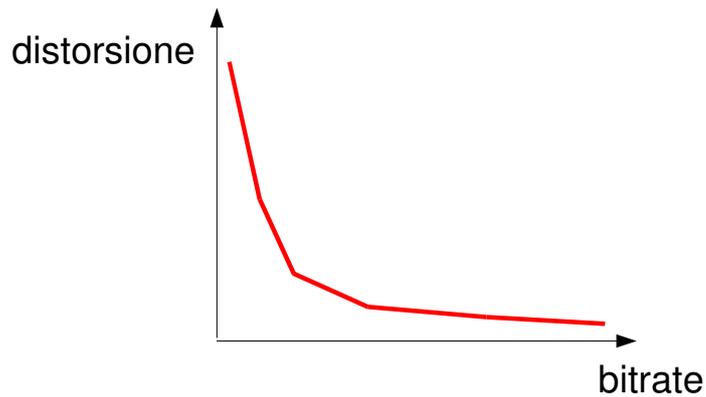
## Distorsione di quantizzazione

- Una possibile misura di distorsione

$$MSE = \frac{\sum_1^N (v_i - v'_i)^2}{N}$$

•  $v_i$ : simbolo prima della quantizzazione

•  $v'_i$ : simbolo dopo la decodifica



21

## Compressione senza perdita di informazione

22

## Sommario

- Eliminazione della correlazione statistica
- Codifica entropica

## Premessa

- Quanto si dirà in questi lucidi si applica a sorgenti di informazioni che producono **serie di dati discreti a precisione finita**:
  - dati digitali su file (es., eseguibili, database)
  - sorgenti multimediali dopo campionamento e quantizzazione
- Le tecniche che verranno presentate **non** provocano una perdita irreversibile di informazione.

## Definizioni

- Una sorgente emette una **serie ordinata di dati**
- Tale serie di dati si può rappresentare come una **sequenza di simboli** dove ciascun simbolo appartiene ad certo **alfabeto** dipendente dal tipo di applicazione
  - Es: campioni sonori quantizzati su 16 bit con segno  
--> l'alfabeto è costituito dai numeri interi nell'intervallo [-32768, 32767]
  - Es: immagine in bianco e nero con pixel su 256 livelli di grigio  
--> l'alfabeto è costituito dai numeri interi nell'intervallo [0, 255]

25

## Cosa vuol dire compressione ?

- Premesso che ogni simbolo deve essere codificato con dei bit per essere elaborato o trasmesso
- Definizione teorica di compressione

*Cambio reversibile di alfabeto tale che la rappresentazione binaria della sequenza di simboli, tradotta nei simboli del nuovo alfabeto, richiede meno bit per essere memorizzata/trasmessa.*

26

# Eliminazione della correlazione statistica

27

## Correlazione statistica

- Sia data una sequenza ordinata di simboli  $s_0, s_1, s_2, \dots, s_n$  di un certo alfabeto emessa da una sorgente.
- C'è correlazione quando la probabilità che  $s_i$  assuma un certo valore dipende dal valore assunto dagli elementi precedenti della sequenza.
- Si dice che l'informazione contenuta nella sequenza è **ridondante** e si può cambiare alfabeto in modo da usare meno bit.

28

## Esempio di correlazione nei testi

- 13% di lettere eliminate
- quante parole si riesce a ricostruire ?

L sp zie s no al eate pre iose nella  
p epa azi ne di suc ulenti pia ti. Es e  
s no ben con sci te fin dall'an ichi à;  
at or o ad es e vi er no ve e e pro rie  
vie com erci li e per esse si  
c mba terono du e bat agl e.

29

## Esempio di correlazione nei testi

- 6% di lettere eliminate
- quante parole si riesce a ricostruire ?

L sp zie sono al eate pre iose nella  
prepa azione di suc ulenti pia ti. Es e  
sono ben conosciute fin dall'antichità;  
at orno ad es e vi erano vere e proprie  
vie com erciali e per esse si  
comba terono dure bat aglie.

30

## Esempio di correlazione nei testi

Le spezie sono alleate preziose nella preparazione di succulenti piatti. Esse sono ben conosciute fin dall'antichità; attorno ad esse vi erano vere e proprie vie commerciali e per esse si combatterono dure battaglie.

31

## Correlazione nelle immagini

- Immagine b/n su due livelli di grigio (fax)

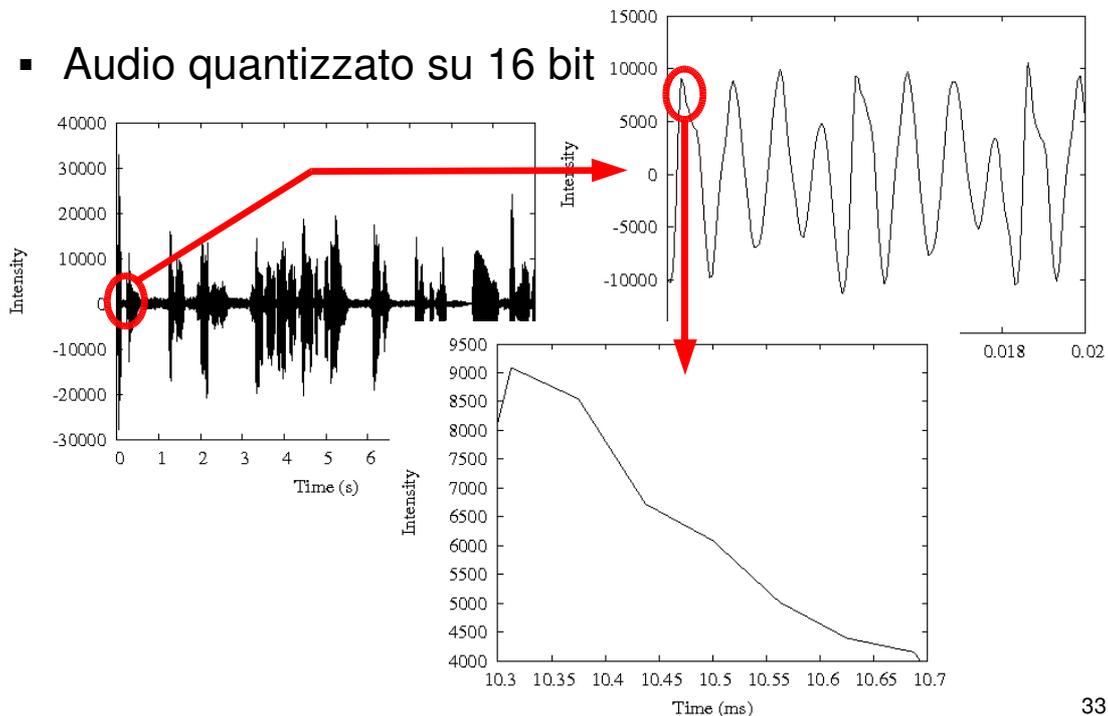


Sequenza: 1,1,1,1,0,0,0,0,0,0,1,1,1,1,1

32

## Correlazione nella voce

- Audio quantizzato su 16 bit



33

## Eliminazione della correlazione

- **Si cambia alfabeto di rappresentazione**
- Codifica run-length
  - Es del fax  
 $1,1,1,1,0,0,0,0,0,0,1,1,1,1,1 \rightarrow (1,4),(0,6),(1,5)$
- Codifica differenziale
  - Es. dell'audio (vedi la figura più dettagliata)  
 $s_0 s_1 s_2 \dots s_n \rightarrow (s_1 - s_0), (s_2 - s_1), \dots, (s_n - s_{n-1})$
- Trasformata
  - Es. dell'audio (vedi la figura intermedia)  
 Sviluppo in serie di Fourier: ampiezza delle componenti sinusoidali  
 $s_0 s_1 s_2 \dots s_n \rightarrow a_0, a_1, \dots, a_n$

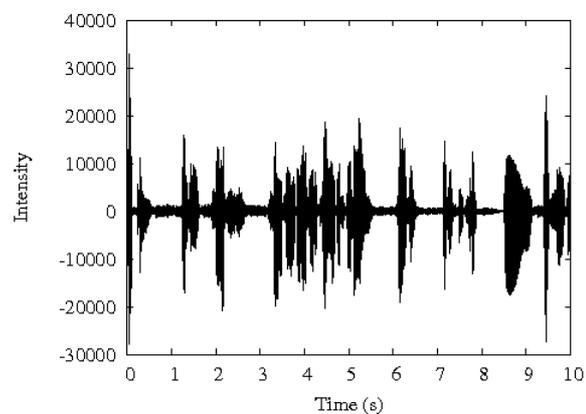
34

# Codifica entropica

35

## Frequenza dei simboli: esempio

- Audio su 16 bit: alfabeto = interi su  $[-32768, 32767]$



- I simboli hanno tutti la stessa probabilità di apparire ?
- Ricordate il caso dell'alfabeto Morse ?

36

## Quanti bit/simbolo ?

- E' meglio rappresentare con meno bit i simboli più frequenti e con più bit quelli meno frequenti.
- Numero minimo teorico di bit/simbolo per l'alfabeto  $s$ :

$$H(s) = \sum_k p_k \log_2 \frac{1}{p_k}$$

- $p_k$  è la frequenza di apparizione del simbolo  $k$ -esimo
- **entropia della sorgente di simboli** (*Shannon*)
- tale formula fornisce un valore medio minimo ma non dice come assegnare i bit ai simboli per ottenerlo !

37

## Esempio di calcolo dell'entropia

- Una sorgente emette simboli appartenenti ad un alfabeto di 4 simboli  $a, b, c, d$
- Frequenze di apparizione dei 4 simboli sono:
  - $1/2, 1/4, 1/8, 1/8$
  - somma delle frequenze uguale a 1
- Entropia della sorgente:

$$\begin{aligned} H(s) &= \frac{1}{2} \log_2(2) + \frac{1}{4} \log_2(4) + \frac{1}{8} \log_2(8) + \frac{1}{8} \log_2(8) \\ &= \frac{1}{2} + \frac{1}{2} + \frac{3}{8} + \frac{3}{8} \\ &= 1.75 \text{ bit / simbolo} \end{aligned}$$

38

## Codifica entropica: problema 1

- Simboli rappresentati su un numero variabile di bit



- come fa il decoder a sapere quando inizia/finisce un simbolo ?

39

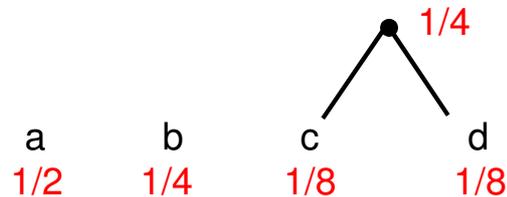
## Algoritmo di Huffman

- Metodo di assegnamento bit ai simboli che garantisce la corretta decodifica
- Si costruisce un albero binario dove le foglie rappresentano i simboli
- Ad ogni biforcazione la somma delle frequenze dei simboli del sotto-albero di destra deve essere il più possibile vicina alla somma delle frequenze dei simboli del sotto-albero di sinistra.
- Ciascun arco di una biforcazione è etichettato con 0 e 1 rispettivamente

40

## Creazione codice di Huffman

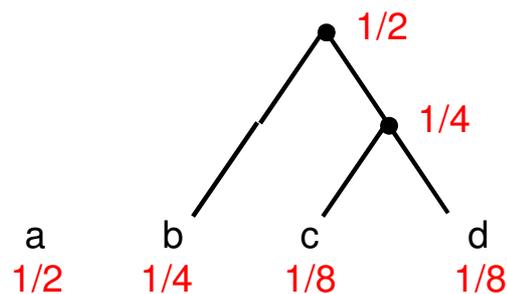
- Alfabeto di 4 simboli  $a, b, c, d$  con frequenze rispettivamente di  $1/2, 1/4, 1/8, 1/8$
- Primo passo:  $a, b, c, d$   
 $1/2 \quad 1/4 \quad 1/8 \quad 1/8$



41

## Creazione codice di Huffman

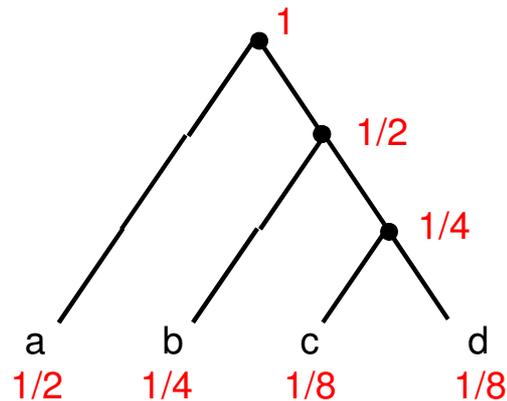
- Alfabeto di 4 simboli  $a, b, c, d$  con frequenze rispettivamente di  $1/2, 1/4, 1/8, 1/8$
- Secondo passo:  $a, b, c, d$   
 $1/2 \quad 1/4 \quad 1/4$



42

## Creazione codice di Huffman

- Alfabeto di 4 simboli  $a, b, c, d$  con frequenze rispettivamente di  $1/2, 1/4, 1/8, 1/8$
- Terzo passo:  $a, b, c, d$   
 $1/2 \quad 1/2$

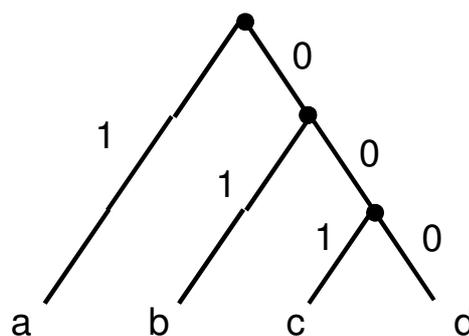


43

## Creazione codice di Huffman

- Alfabeto di 4 simboli  $a, b, c, d$  con frequenze rispettivamente di  $1/2, 1/4, 1/8, 1/8$
- Creazione del codice:

a --> 1  
b --> 01  
c --> 001  
d --> 000

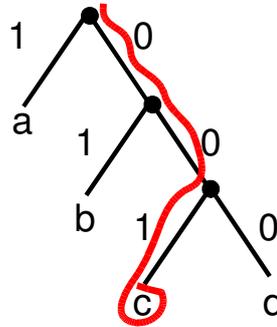


44

## Decodifica codice di Huffman

- In ricezione per decodificare i simboli si visita l'albero binario in base ai bit in arrivo

Es: 001 ---> c



45

## Huffman: considerazioni

- L'algoritmo di Huffman è ottimo (rispetto all'entropia) se le frequenze sono potenze di 2.
- Per distribuzioni di frequenze più complicate esistono algoritmi più sofisticati
  - Codifica aritmetica
  - Codici di Golomb

46

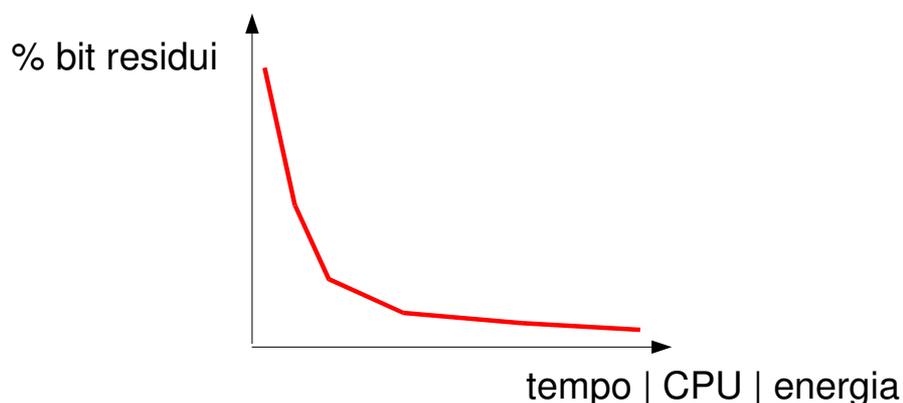
## Codifica entropica: altri problemi

- Stima delle frequenze  $p_k$ 
  - migliore è la stima e maggiore è la compressione
  - ho bisogno di fare ipotesi vincolanti sulla sorgente oppure considerare un elevato numero di esempi
- Errori sul bit durante la trasmissione rendono la decodifica impossibile
  - punti di re-sincronizzazione periodici nel flusso di bit
  - diminuzione dell'efficienza di compressione

47

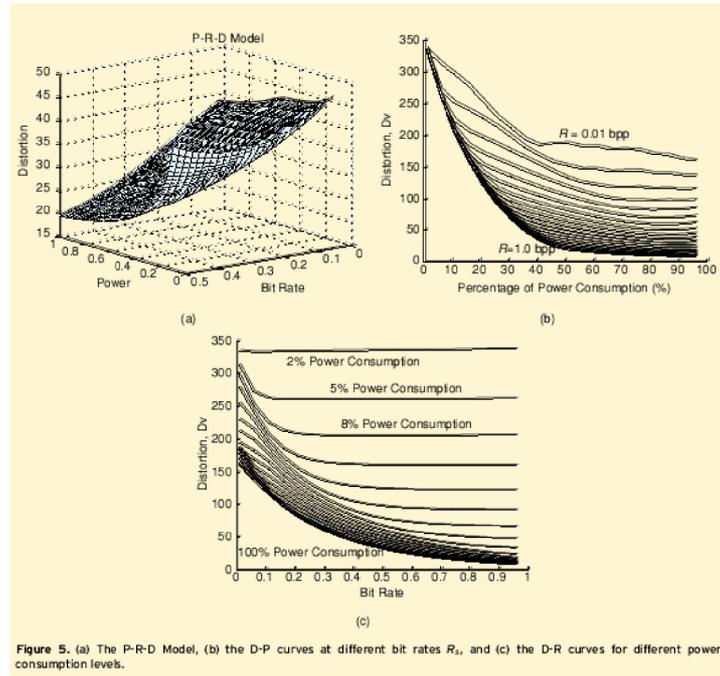
## Considerazioni sulla complessità

In generale se si vuole comprimere di più una sequenza di simboli si devono usare **algoritmi più complessi** che impiegano più tempo oppure necessitano di una **CPU più potente** e comunque sempre portano ad un **maggiore consumo di energia**.



48

## Considerazioni sulla complessità



49

**Principali algoritmi di  
compressione  
per voce, immagini e video**

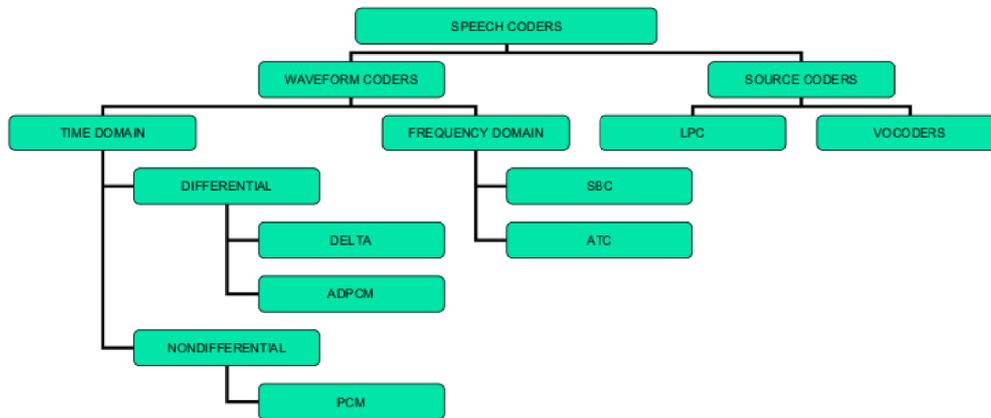
50

## **Sommario**

- Compressione della voce
- Compressione di immagini
  - JPEG
- Compressione di video
  - algoritmi
  - principali standard

## **Compressione della voce**

## Principali metodi compressione voce



Courtesy of R. Z. Zaputowycz

53

## Pulse Code Modulation (PCM)

- Il segnale vocale viene
  - filtrato a 3.3 kHz
  - campionato a 8 kHz
  - quantizzato su 8 bit
- Quantizzatore logaritmico
  - preciso a volume basso
  - approssimato a volume alto
  - adatto alla percezione umana
- ITU-T G.711
  - 64 kb/s
  - due tipi di curve logaritmiche: uLaw, aLaw

54

## Differential PCM (DPCM)

- Il segnale vocale viene
  - filtrato a 3.3 kHz
  - campionato a 8 kHz
  - quantizzato su 16 bit
- Codifica della differenza tra due campioni successivi
- Eliminazione della correlazione statistica

55

## Adaptive Differential PCM (ADPCM)

- Il segnale vocale viene
  - filtrato a 3.3 kHz
  - campionato a 8 kHz
  - quantizzato su 16 bit
- Codifica della differenza tra due campioni successivi
- Quantizzazione della differenza variabile nel tempo (**adattativa**)
  - bassa precisione quando il volume e' alto
  - alta precisione quando il volume e' basso
- ITU-T G.726
  - vari bitrate: 40, 32, 24, 16 kb/s

56

# Compressione di immagini

57

## Immagini in bianco e nero

- Campionamento
  - Griglia di pixel
  - Risoluzione data da colonne x righe
- Quantizzazione
  - 8 bit/pixel --> Matrice di valori interi su 8 bit
  - La quantizzazione può essere uniforme o non-uniforme (p.es., logaritmica), a seconda della distribuzione di probabilità delle intensità e il comportamento percettivo umano.

58

## Immagini a colori

- Ogni colore è dato dalla sovrapposizione di 3 componenti
- Tricromia additiva
  - R (rosso)
  - G (verde)
  - B (blu)
- Tre matrici per rappresentare la griglia di pixel

59

## Rappresentazione YUV

- In principio la TV era solo in bianco e nero
- Poi avvio delle trasmissioni a colori
  - necessità di poter vedere le trasmissioni a colori anche su televisori in bianco e nero
- Passaggio da rappresentazione RGB a YUV:
  - Y, luminanza (toni di grigio)
  - U e V cromaticità
- I televisori in bianco e nero usano solo la componente Y
- I televisori a colori decodificano anche U e V

60

## Conversione RGB --> YUV

- La componente Y è una media pesata delle componenti RGB
  - $Y = r R + g G + b B$
- $U = B - Y$
- $V = R - Y$
- La conversione RGB --> YUV è una trasformazione lineare
  - reversibile
  - rappresentazione matriciale

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} r & g & b \\ -r & -g & 1-b \\ 1-r & -g & -b \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

61

## Percezione delle comp. YUV

- L'occhio umano è molto sensibile ai contorni
  - I contorni sono veicolati dalla componente Y
- ↓
- Le componenti U e V possono avere una risoluzione spaziale minore della componente Y senza evidenti perdite di qualità
- ↓
- Opportunità di compressione (lossy)

62

## Formato CIF

Base				3 matrici 352 x 288
4:4:4				3 matrici 352 x 288
4:2:2				1 matrice 352 x 288 e 2 matrici 176 x 288
4:1:1				1 matrice 352 x 288 e 2 matrici 176 x 144

63

## Formato Quarter-CIF (QCIF)

Base				3 matrici 176 x 144
4:4:4				3 matrici 176 x 144
4:2:2				1 matrice 176 x 144 e 2 matrici 88 x 144
4:1:1				1 matrice 176 x 144 e 2 matrici 88 x 72

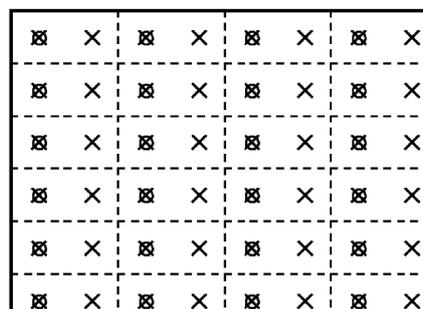
64

## Sottocampionamento di U e V

- Prima di sottocampionare le componenti U e V bisogna passarle in un filtro passa-basso al fine di ridurre la larghezza e soddisfare il teorema di Nyquist
- ESEMPIO DIMOSTRATIVO: supponiamo di dover sottocampionare del 50% un'immagine con righe bianche e nere alternate:
- filtrando ottengo un'immagine grigia (OK)
- senza il sottocampionamento avrei un'immagine tutta nera o tutta bianca (**aliasing**)

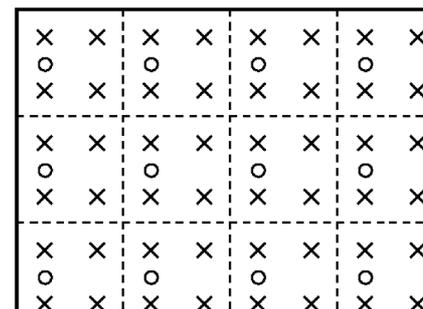
65

## Sottocampionamento: campioni e pixel



X Represent luminance samples  
O Represent chrominance samples

4:2:2



X Represent luminance samples  
O Represent chrominance samples

4:1:1

66

## Sottocampionamento: guadagno

- Formato QCIF
- 8 bit per campione (non per pixel !)
- **4:4:4**
  - $176 \times 144 \times 3 = 76464$  byte
- **4:2:2**
  - $176 \times 144 \times 2 = 50976$  byte
- **4:1:1**
  - $176 \times 144 \times 1,5 = 38232$  byte

67

## Sottocampionamento: applicazioni

- **4:4:4**
  - applicazioni studio, mediche e per archiviazione
  - utile quando occorre elaborare più volte l'immagine
- **4:2:2 e 4:1:1**
  - sufficiente per applicazioni domestiche
- **ATTENZIONE**

La compressione 4:2:2 e 4:1:1 si può effettuare solo su matrici YUV

68

## JPEG: passi

- Conversione RGB --> YUV
- Sottocampionamento 4:4:4 --> 4:1:1 (**lossy**)
- Eliminazione della correlazione statistica mediante Discrete Cosine Transform (DCT) su blocchi di 8x8 campioni
  - 16x16 pixel --> 4 blocchi Y, 1 U e 1 V
- Quantizzazione dei coefficienti DCT (**lossy**)
- Codifica run-length
- Codifica di Huffman delle coppie run-length

69

## Discrete Cosine Transform

- Permette il passaggio dal dominio spaziale a quello delle frequenze spaziali.
  - Sottocaso dello Sviluppo in Serie di Fourier
- In forma matematica, da matrice  $y$  a matrice  $z$  via moltiplicazione per la matrice che definisce la trasformata
  - $z = A y$  trasformazione diretta
  - $y = A^{-1} z$  trasformazione inversa
  - matrice  $z$ : matrice 8x8 dei coefficienti delle componenti sinusoidali a varie frequenze
- Operazione **lineare** e **reversibile**
  - in teoria non introduce distorsione (**lossless**)
  - in pratica arrotondamento:  $y$  e  $z$  sono interi,  $A$  è float

70

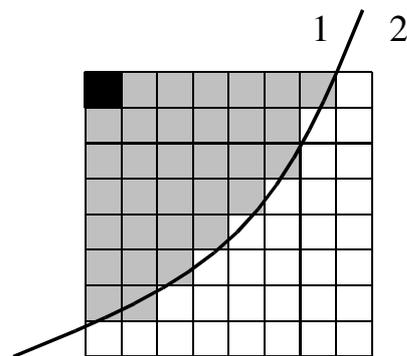
## DCT su immagini

- Primo coefficiente --> frequenza nulla --> valor medio di luminosità (oppure U o V)
- Segnale poco ripido --> coefficienti nulli a frequenze elevate
- Segnale ripido --> alti coefficienti a frequenze elevate
- Le immagini naturali contengono molte zone di colore piatte o sfumate
  - molto spesso i coefficienti a frequenze elevate sono nulli
- I contorni degli oggetti introducono alti coefficienti a frequenze elevate

71

## Matrice trasformata

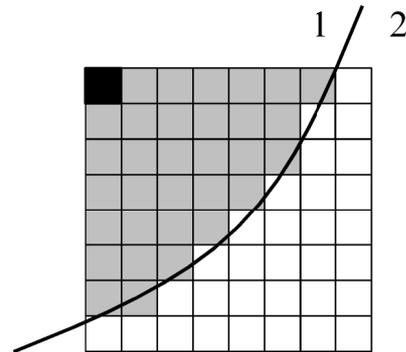
- La cella nera contiene la componente continua
- Le sfumature contribuiscono ai coefficienti nell'area 1
- I bordi invece ai coefficienti nell'area 2
- Spesso, per immagini naturali, l'area 2 contiene molti coefficienti piccoli o nulli.



72

## Quantizzazione dei coefficienti DCT

- L'occhio è poco sensibile alla precisione dei coefficienti alle alte frequenze spaziali
- Tutti i coefficienti vengono divisi per delle costanti note al decodificatore
- Al risultato della divisione viene troncata la parte decimale
- Quantizzazione lineare

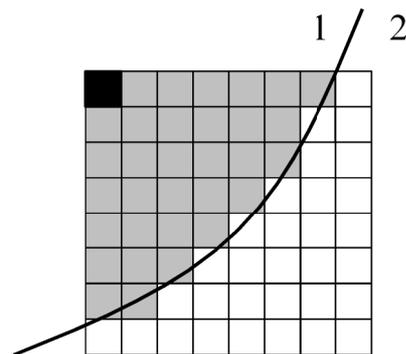


8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

73

## Quantizzazione dei coefficienti DCT

- I già bassi valori dei coefficienti nell'area 2 diventano spesso nulli
- Maggiore è il divisore e maggiore è l'errore di quantizzazione (irreversibile)
- Introduzione di distorsione (effetto blocchi delle immagini JPEG molto compresse)



8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

74

## Effetto della quantizzazione dei coefficienti DCT



alto bitrate

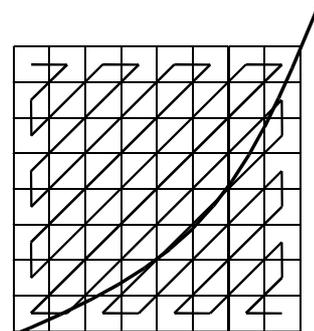


basso bitrate

75

## Codifica Run-length dei coefficienti

- Percorso seguito per la scansione dei coefficienti
- Massimizza la probabilità di trovare sequenze di valori uguali
- Dopo la metà del percorso è molto probabile che ci sia un'unica sequenza di zeri che diventa un unico simbolo run-length (simbolo ad alta probabilità)



76

## Codifica di Huffman

- Le coppie run-length (valore del coefficiente, ripetizioni) vengono codificate con codici binari a lunghezza variabile.
- Le coppie più frequenti sono rappresentate da codici più corti.

77

## Compressione del video

78

## Ridondanza spaziale e temporale

- Il Video si può assimilare ad una sequenza di immagini dette **frame**
  - in genere 20-30 frame/s per qualità film.
- La forma di codifica video più semplice consiste nel codificare i frame come immagini indipendenti
  - MotionJPEG --> ogni frame è un JPEG
- Nei video oltre alla correlazione SPAZIALE (DCT – JPEG) esiste anche la correlazione TEMPORALE (cioè tra frame diversi)
- Per eliminare la correlazione temporale si usa la **codifica differenziale con compensazione del moto**

79

## Codifica differenziale con compensazione del moto

- Il primo frame viene codificato come in JPEG
  - frame indipendente e autoreferenziante
  - frame INTRA (**I-frame**).
- Per ciascun blocco 16 x 16 pixel (**macroblocco**)
  - Ricerca nel frame precedente dell'area d'immagine più simile al macroblocco (**motion estimation**)
- Codifica INTRA (come in JPEG) della differenza tra le 2 matrici (**residuo**) e codifica della posizione relativa tra le due aree (**vettore di moto**).

80

## Motion estimation

- E' una fase computazionalmente intensa
  - consumo di tempo o necessita' di CPU potente
  - consumo di energia
- Si usano soluzioni SUB-OTTIME:
  - si riduce il range di ricerca
  - si compie la ricerca su versioni sottocampionate dei 2 frame (e quindi di dimensioni ridotte)

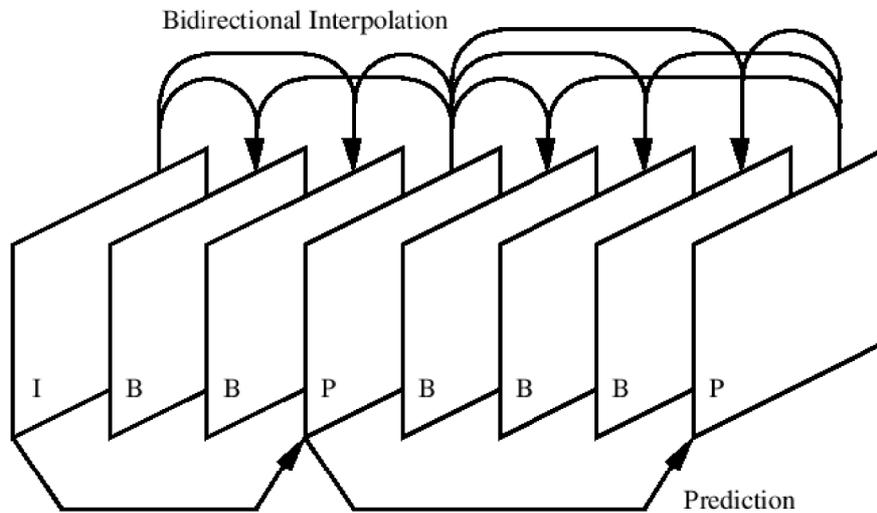
81

## Tipi di frame

- Frame di tipo INTRA ( I )
  - codifica JPEG indipendente dai frame vicini
- Frame di tipo PREDITTIVO ( P ).
  - codifica differenziale con compensazione del moto rispetto ad un frame precedente
  - dipendente da tale frame
- Frame di tipo BI-DIRECTIONAL ( B ).
  - codifica differenziale con compensazione del moto rispetto ad un frame precedente e ad uno successivo
  - aumenta la probabilità di avere residui bassi
  - dipendente da entrambi i frame
  - alta intensita' computazionale

82

## Tipi di frame (2)



83

## Display vs. Coding Order

- Sequenza di frame come viene catturata o riprodotta (**display order**):

I B P B P B P B P ..... I

Frame # 0 1 2 3 .....

- Ordine di codifica e trasmissione dei frame (**coding order**):

Frame # 0 2 1 .....

- Si Codifica il frame 0, poi il frame 2, dopodichè avendo informazioni su 0 e 2 si puo' codificare il frame 1 di tipo B. --> **RITARDO ALGORITMICO**

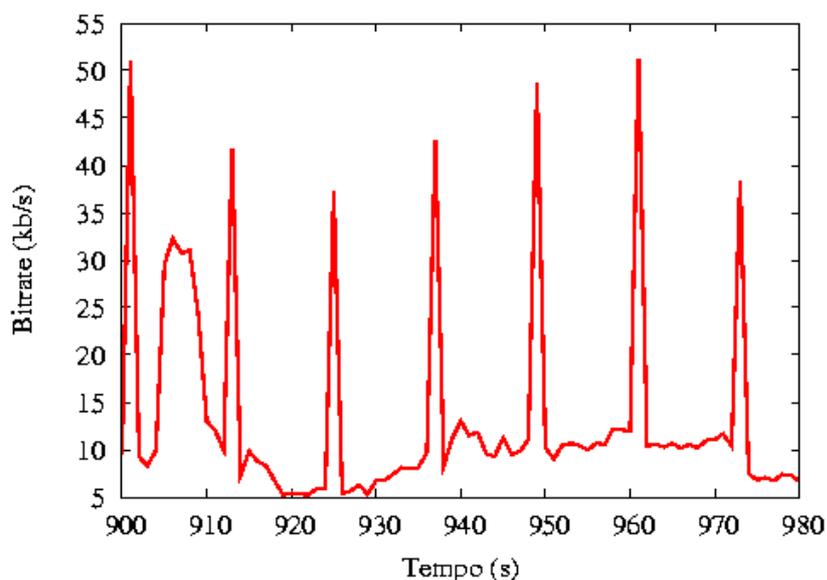
84

## Andamento del bitrate

- I frame di tipo I hanno bit-rate più elevato rispetto agli altri perché non implementano la compensazione del moto.
- I frame di tipo B invece hanno il bit-rate più basso perché implementano una doppia compensazione del moto.
- I frame di tipo P hanno una dimensione intermedia.
- La trasmissione di video compresso genera traffico VBR.

85

## Trasmissione di video compresso



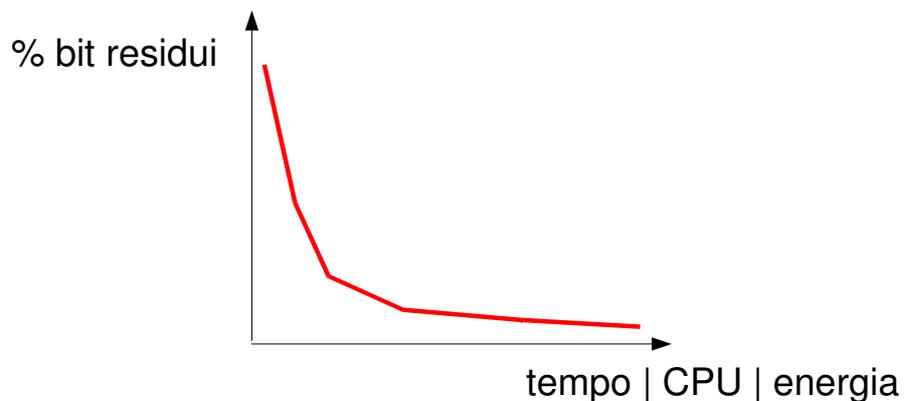
86

## Complessità computazionale

- E' il reciproco dell'andamento del bit-rate !
- I frame di tipo I hanno ridotta complessità perché non c'è compensazione del moto.
- I frame di tipo B invece hanno elevata complessità perché effettuano una doppia compensazione del moto.
- I frame di tipo P sono una via di mezzo (effettuano una sola compensazione del moto).
- Occorre tenere conto di tale variabilità nel progetto dell'architettura del codificatore (hardware, software).

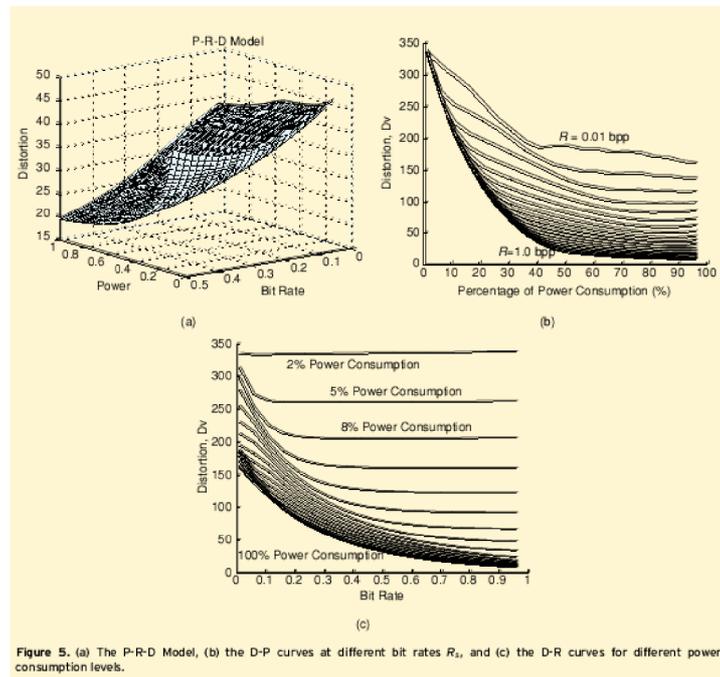
87

## Complessità computazionale (2)



88

## Complessità computazionale (3)



89

## Ritardo

- Il ritardo è MINIMO per tutti i frame di tipo I
- MEDIO per i frame I + P
- ALTO per i frame I + P + B dovuto al ritardo algoritmico (quindi indipendente dalla potenza della CPU !)
- Per applicazioni interattive (videoconferenza) si usano solo frame I e P
- Frame di tipo B si usano per lo streaming

90

## Standard di compressione video

- ITU-T H.261
  - videoconferenza
- ISO MPEG-1
  - VideoCD (~1.4 Mb/s)
- ISO MPEG-2 / ITU-T H.262
  - DVD (~3 Mb/s)
  - DVB: digitale terrestre e satellite (~2 Mb/s)
- ITU-T H.263
  - videoconferenza
- ISO MPEG-4
  - DivX, XviD (700 kb/s – 5 Mb/s quasi lossless)
- ITU-T H.264 (AVC oppure MPEG4/10)
  - **stato dell'arte !!!**