



Politecnico
di Milano

Rappresentazione dell'informazione

Corsi di Informatica Grafica

Prof. Manuel Roveri

Dipartimento di Elettronica e Informazione
roveri@elet.polimi.it



Indice

- Rappresentazione dell'informazione
 - Numeri
 - Caratteri e testi
 - Suoni
 - Immagini
 - Video



Codifica dati e istruzioni

- **Algoritmi = istruzioni + dati.**
- Per scrivere un programma che descriva un algoritmo è necessario rappresentare **istruzioni** e **dati** utilizzando un formato che il calcolatore sia in grado di:
 - Memorizzare
 - Elaborare
 - Trasmettere



Codifica dati e istruzioni

- **Alfabeto dei simboli**
 - Esempio: cifre 0, 1
 - Esempio: cifre 0, 1, 2, ..., 9, separatore decimale (.), separatore delle migliaia (.), segno positivo (+) e negativo (-).
- **Regole di composizione (sintassi)** che definiscono le sequenze di cifre (**parole**) ammissibili:
 - Esempio: 1.234,5 è la rappresentazione di un numero reale.
 - Esempio: 1,23,45 non lo è.



Codifica dati e istruzioni

- **Codice (semantica):** insieme di regole che ad ogni configurazione ammissibile associa un'entità di informazione
 - $1.234,5 = 1 \times 10^3 + 2 \times 10^2 + 3 \times 10^1 + 4 \times 10^0 + 5 \times 10^{-1}$
 - $1,23,45 = ?$
- Lo stesso alfabeto può essere utilizzato con codici diversi:
 - $123,456 = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 + 4 \times 10^{-1} + 5 \times 10^{-2} + 6 \times 10^{-3}$ **(IT)**
 - $123,456 = 1 \times 10^5 + 2 \times 10^4 + 3 \times 10^3 + 4 \times 10^2 + 5 \times 10^1 + 6 \times 10^0$ **(UK)**



Codifica binaria dell'informazione

- Il calcolatore utilizza un **alfabeto binario**: usiamo dispositivi elettronici digitali in grado di assumere due solo stati: acceso/spento, ON/OFF, I/0, VERO/FALSO.
- Un alfabeto binario non limita le funzionalità di un calcolatore.
- Il simbolo o cifra binaria si indica con **bit** (da **Binary digIT**): quantità d'informazione che si ottiene selezionando una configurazione da un insieme che ne contiene due.
- La risposta **SI** oppure **NO** a una domanda porta **1 bit** di informazione.



Codifica binaria dell'informazione

- Il calcolatore tratta diversi tipi di dati (numeri, caratteri, ecc.) tutti rappresentati con la codifica binaria.
- **Problema:** assegnare un codice univoco a tutti gli oggetti compresi in un insieme predefinito.
- Esempio: associare un codice binario ai giorni della settimana.



Codifica binaria dell'informazione

- **Quanti oggetti** diversi posso codificare con parole binarie composte da **k bit**?
 - 1 bit: $2^1 = 2$ stati (0, 1) $\Rightarrow$ 2 oggetti
 - 2 bit: $2^2 = 4$ stati (00, 01, 10, 11) $\Rightarrow$ 4 oggetti
 - 3 bit: $2^3 = 8$ stati (000, 001, 010, 011, 100, 101, 110, 111) $\Rightarrow$ 8 oggetti
 - ...
 - k bit: 2^k stati $\Rightarrow 2^k$ oggetti



Codifica binaria dell'informazione

- Se passiamo da una parola binaria di **k bit** ad una parola di **$k+1$ bit** si raddoppia il numero di oggetti che si possono rappresentare (**$2^k + 1$**).



Codifica binaria dell'informazione

- **Quanti bit** mi servono per codificare N oggetti:
 - $N \leq 2^k \Rightarrow k \geq \log_2 N \Rightarrow \mathbf{k = \lceil \log_2 N \rceil}$
- Ipotesi implicita: le parole di un codice hanno tutte la stessa lunghezza.



Definire un codice

- Identificare due insiemi:
 - Insieme delle configurazioni ammissibili;
 - Insieme degli oggetti da rappresentare.
- Associare gli elementi dei due insiemi



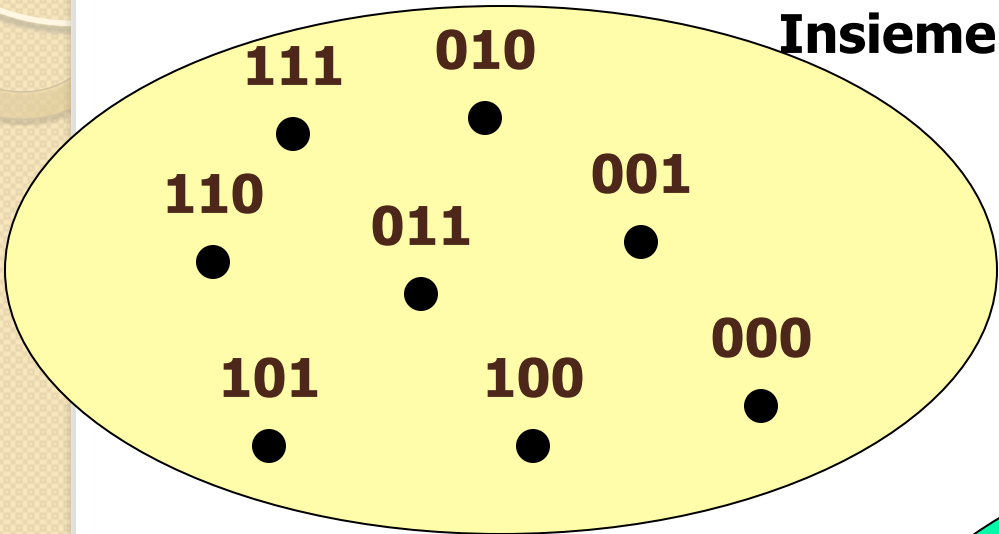
Esempio

- Associare una codifica binaria ai giorni della settimana.
- Quanti bit devono avere le parole binarie usate per identificare i giorni della settimana (7 oggetti diversi)?
 - $k = \lceil \log_2 7 \rceil = 3$

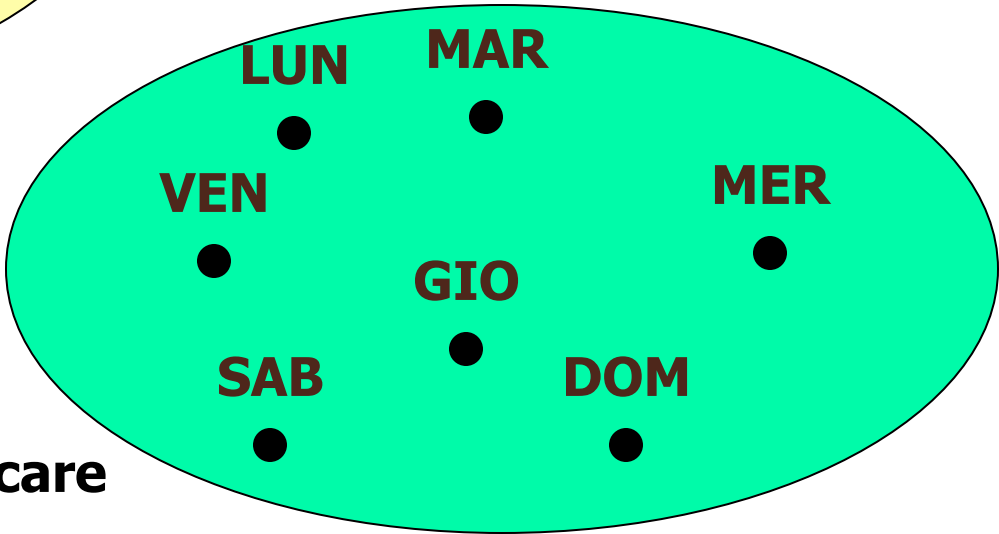


Esempio

Insieme delle configurazioni ammissibili

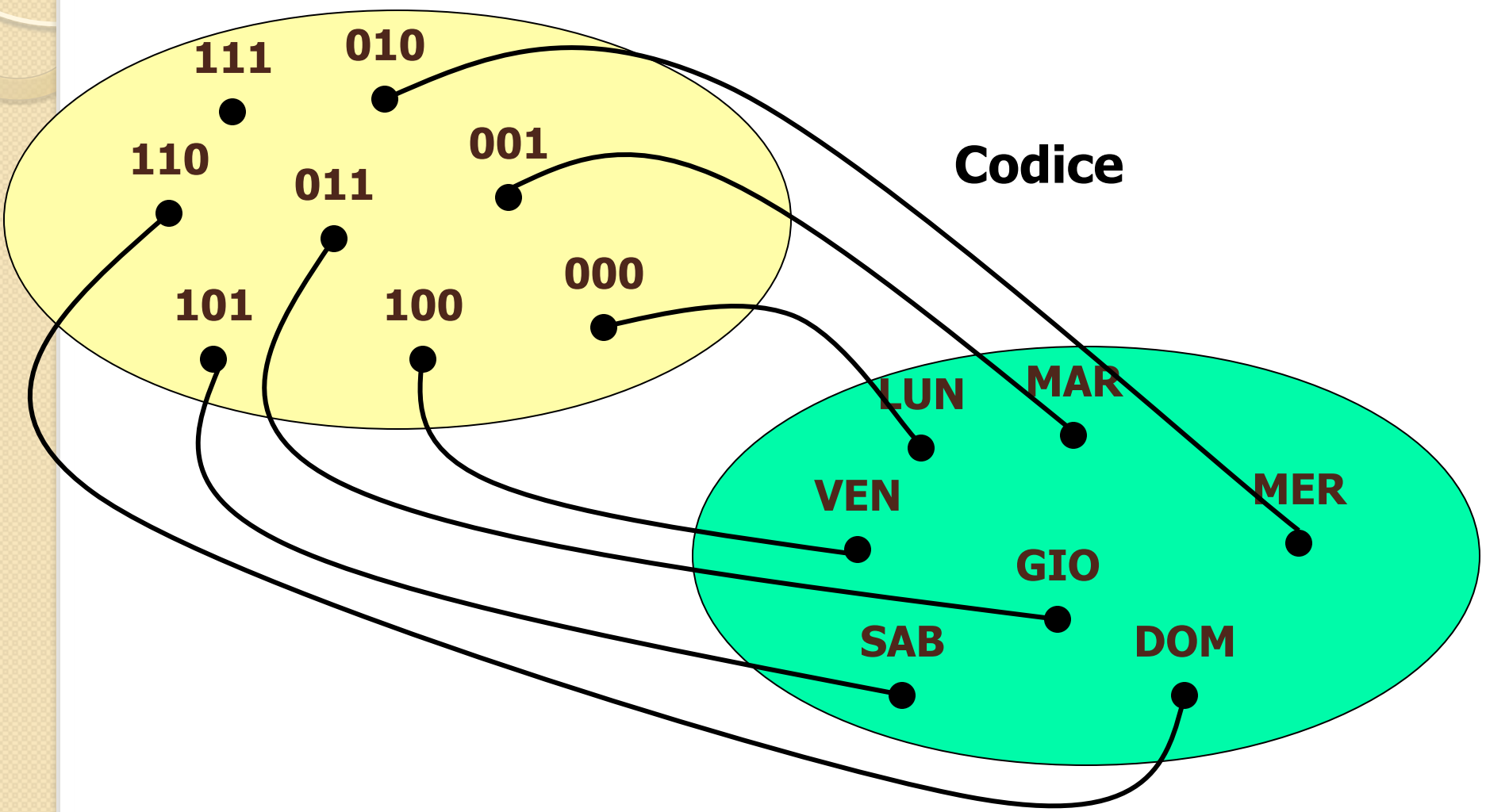


Insieme degli oggetti da codificare





Esempio





Byte

- **bit** rappresenta 2 stati: 0 oppure 1.
- **Byte** rappresenta 8 bit $\Rightarrow 2^8 = 256$ stati
- **KiloByte [KB]** = 2^{10} Byte = 1024 Byte $\approx 10^3$ Byte
- **MegaByte [MB]** = 2^{20} Byte = 1048576 Byte $\approx 10^6$ Byte
- **GigaByte [GB]** = 2^{30} Byte $\approx 10^9$ Byte
- **TeraByte [TB]** = 2^{40} Byte $\approx 10^{12}$ Byte
- ...



Codifica binaria dei caratteri

- Quanti sono gli oggetti da rappresentare?
 - 26 lettere maiuscole
 - 26 lettere minuscole
 - 10 cifre
 - Circa 30 simboli d'interpunzione (, ; ...)
 - Circa 30 caratteri di controllo (EOF, CR, ...)
- Totale circa 120 oggetti complessivi
 $\Rightarrow k = \lceil \log_2 120 \rceil = 7.$



Codifica binaria dei caratteri

- **Codice ASCII** (American Standard Code for Information Interchange) utilizza **7 bit**
⇒ può rappresentare $2^7 = 128$ caratteri detti caratteri ASCII Standard.
- **Codice ASCII esteso** utilizza **8 bit (1 Byte)**
⇒ può rappresentare $2^8 = 256$ caratteri detti caratteri ASCII estesi.
 - Tale codice comprende i caratteri ASCII standard e alcuni caratteri semigrafici (cornici, lettere nazionali, simboli matematici, ecc.)
- **Codice UNICODE** utilizza **16 bit (2 Byte)**



Notazione posizionale

- Consideriamo i numeri interi assoluti (numeri naturali più lo zero).
- **Notazione posizionale in base b:**

Alfabeto: $c_i \in \{0 \dots b-1\}$

$c_k c_{k-1} \dots c_1 c_0$ rappresenta:

$$N = c_k \times b^k + c_{k-1} \times b^{k-1} + \dots + c_0 \times b^0 = \sum_{(i=0 \dots k)} c_i \times b^i$$

c_0 bit meno significativo

c_k bit più significativo



Notazione posizionale

- Basi più frequentemente utilizzate:
base 2 : $c_i \in \{0, 1\}$
base 8: $c_i \in \{0, 1, 2, 3, 4, 5, 6, 7\}$
base 10: $c_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
base 16:
 $c_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$



Codifica posizionale dei numeri: il sistema arabico

- Il sistema di numerazione arabico sfrutta una codifica posizionale
 - Ogni cifra assume un valore particolare a seconda della posizione assunta all'interno del numero
 - Esempio:
01 è diverso da 10
- Il sistema arabico usa 10 cifre
 - Si parla di numerazione in base 10
- Il valore di una cifra α all'interno di un numero è pari a:
 $\alpha \times 10^n$
dove n è la posizione occupata dalla cifra α all'interno del numero (si conta da destra partendo da 0)
 - Esempio:
$$1257 = 7 \times 10^0 + 5 \times 10^1 + 2 \times 10^2 + 1 \times 10^3$$



Codifica binaria

- Numeri binari: base **$b = 2$**
- Alfabeto binario: cifre $c_i \in \{0, 1\}$
- $(c_k \dots c_1 c_0)_2$ rappresenta: **$N = \sum_{(i=0 \dots k)} c_i \times 2^i$**



Conversione binario $\Rightarrow$ decimale

- Dato il numero **N** espresso in **base 2**, basta riscriverlo secondo la notazione posizionale utilizzando cifre e potenze della **base 10**.
- Esempi:

$$N = 1101_2$$

$$N = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13_{10}$$

$$N = 100101_2$$

$$N = 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 37_{10}$$



Conversione decimale $\Rightarrow$ binario

- Si calcolano i resti della divisione per due finché il risultato di una divisione diventa zero:
- Esempio: $N = 23_{10}$

$$23 : 2 = 11$$

$$11 : 2 = 5$$

$$5 : 2 = 2$$

$$2 : 2 = 1$$

$$1 : 2 = 0$$

$$\text{resto } R_0 = b_0 = 1$$

$$\text{resto } R_1 = b_1 = 1$$

$$\text{resto } R_2 = b_2 = 1$$

$$\text{resto } R_3 = b_3 = 0$$

$$\text{resto } R_4 = b_4 = 1$$

$$\xrightarrow{\quad} 23_{10} = 10111_2$$



Conversione decimale $\Rightarrow$ binario

- Esempio: $N = 18_{10}$

$$18 : 2 = 9$$

$$9 : 2 = 4$$

$$4 : 2 = 2$$

$$2 : 2 = 1$$

$$1 : 2 = 0$$

$$\text{resto } R_0 = b_0 = 0$$

$$\text{resto } R_1 = b_1 = 1$$

$$\text{resto } R_2 = b_2 = 0$$

$$\text{resto } R_3 = b_3 = 0$$

$$\text{resto } R_4 = b_4 = 1$$

$$\rightarrow 18_{10} = 10010_2$$



Codifica binaria su n bit

- Con una sequenza di **n bit** si possono rappresentare **2^n** numeri interi assoluti da **0** a **$2^n - 1$** .
- Esempio: **$n = 4$ bit $\Rightarrow 2^4 = 16$** numeri interi assoluti da **0** a **15**.
- La lunghezza delle sequenze di bit adottate stabilisce il massimo numero che può essere rappresentato.



Codifica binaria su $n+1$ bit

- Se passiamo da una sequenza di lunghezza **n bit** ad una sequenza di **$n+1$ bit** si raddoppiano i numeri che si possono rappresentare (2^{n+1}).
- Esempio:
 - **$n = 4$ bit $\Rightarrow 2^4 = 16$ numeri da **0** a **15**.**
 - **$n = 5$ bit $\Rightarrow 2^5 = 32$ numeri da **0** a **31**.**



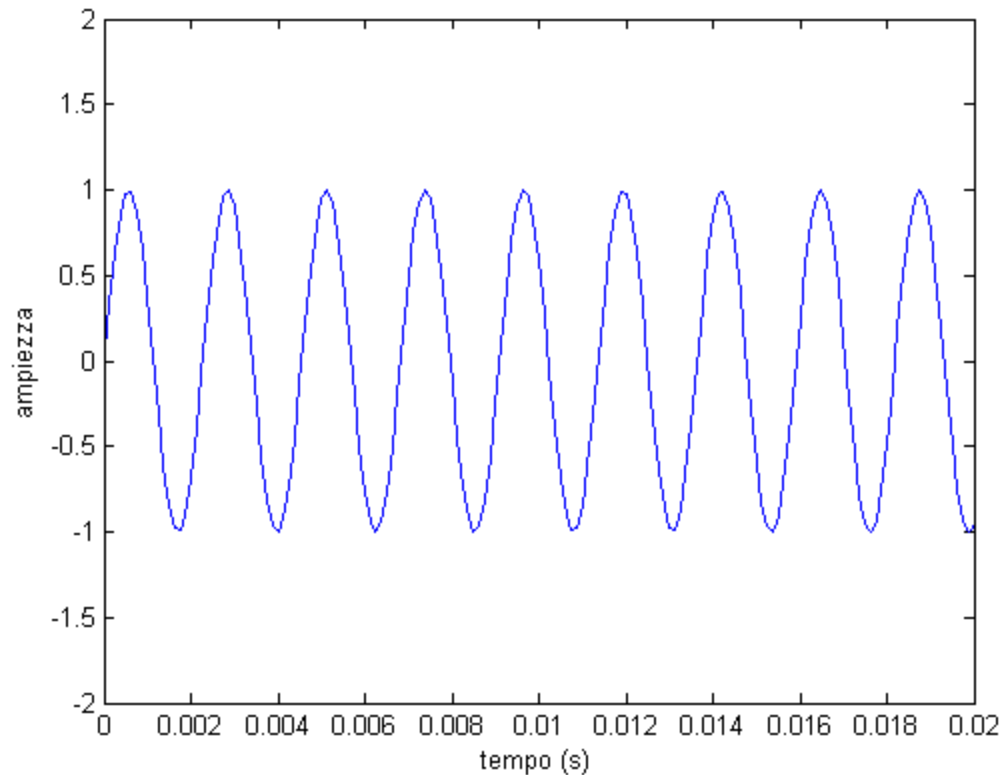
Codifica dei caratteri e delle stringhe

- Per la codifica dei caratteri e delle stringhe (sequenze di caratteri come parole e frasi) si sfruttano delle convenzioni che associano un valore numerico ad ogni carattere
- Le più diffuse convenzioni sono:
 - La convenzione ASCII
 - Sfrutta sette bit (otto nella versione estesa) per rappresentare un massimo di 128 (256) caratteri
 - La convenzione UNICODE
 - Sfrutta 16 bit per rappresentare fino a 65.536 caratteri
 - Utile nel caso di alfabeti particolarmente complessi quale quello cinese



Segnali audio

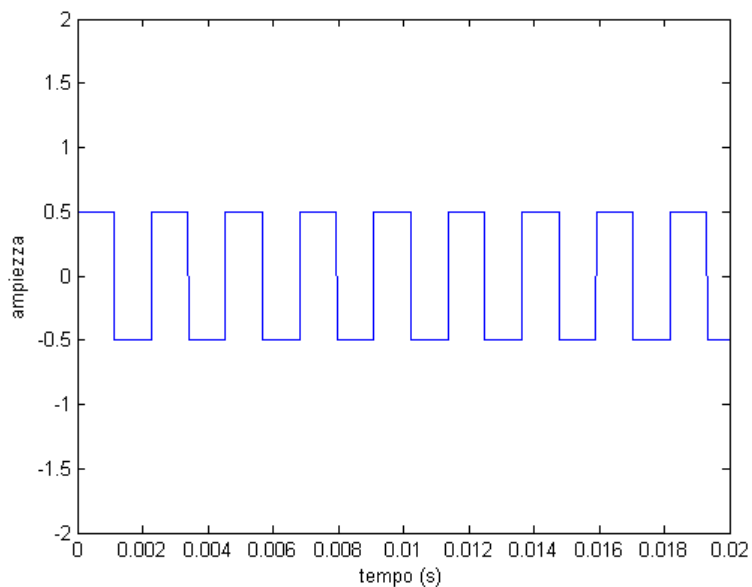
Esempio:
la nota LA



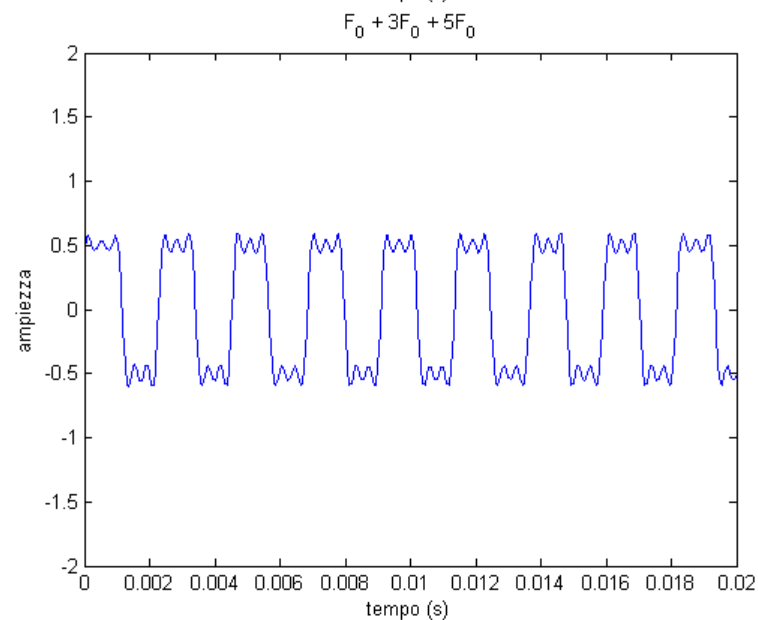
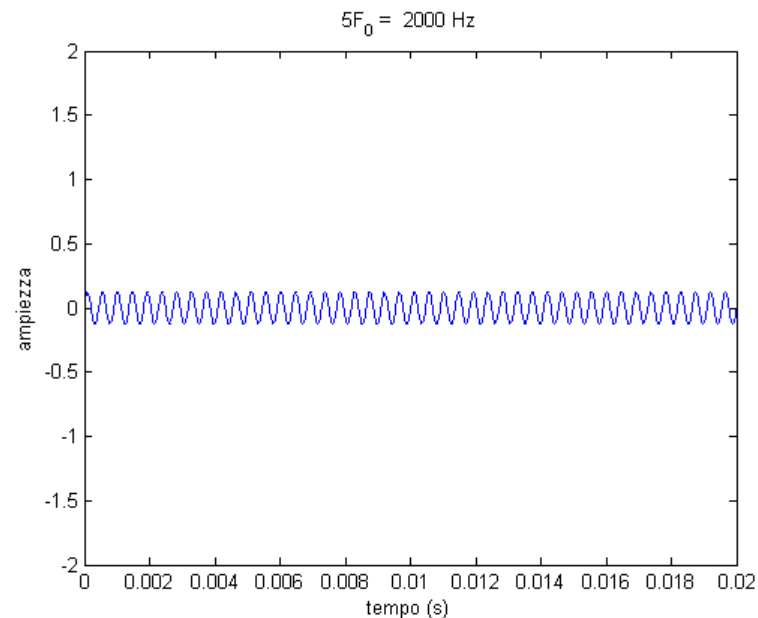
La pressione dell'aria varia in
modo sinusoidale con una
frequenza di 440Hz



Analisi dei segnali audio



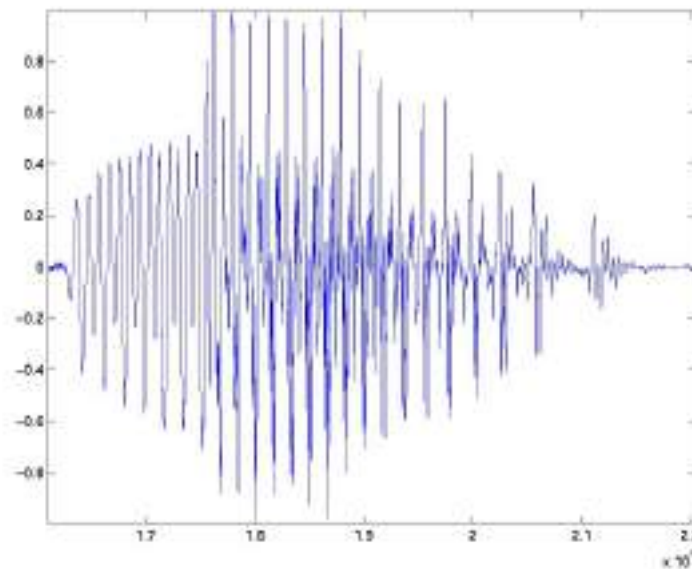
Tutti i suoni possono essere visti
come la somma pesata di suoni
semplici (sinusoidi)





Codifica audio

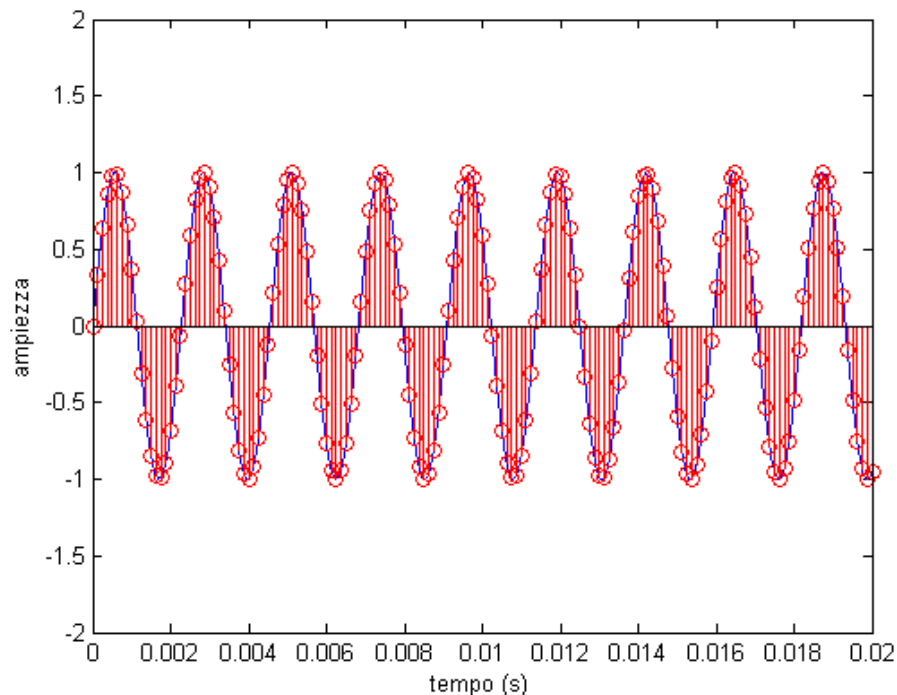
- Onde sonore: segnali ‘continui’
 - Discretizzazione del tempo
 - Campionamento: scelta di istanti in cui considerare il valore del segnale (discretizzazione)
 - Discretizzazione delle ampiezze
 - Quantizzazione: codifica predefinito di bit





Campionamento

- Si misura l'ampiezza del segnale analogico a intervalli regolari, ogni T secondi
- T è detto periodo di campionamento (in secondi)
- $F = 1/T$ è detta frequenza di campionamento (in Hz)





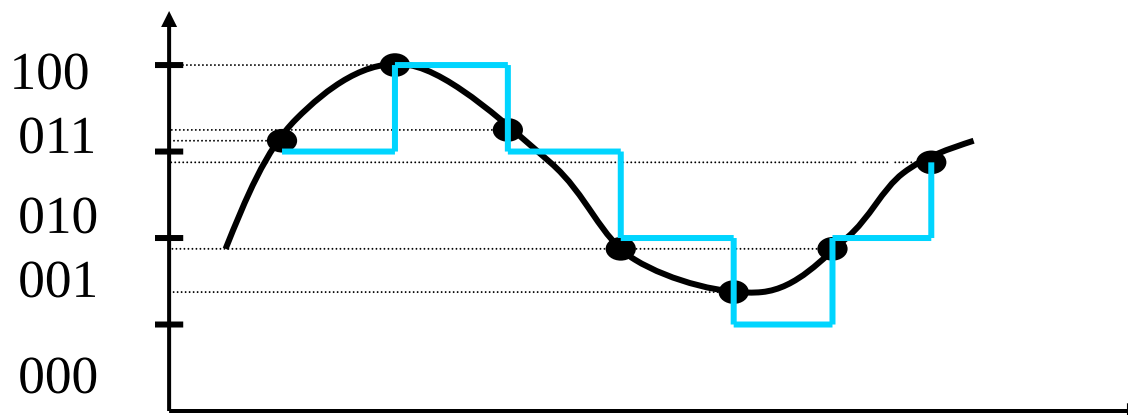
Campionamento

- Per segnali audio di tipo vocale (ad es. telefono), la frequenza di campionamento è tipicamente di 8 kHz
- Per segnali audio musicali (ad es. CD audio), la frequenza di campionamento è tipicamente di 44.1 kHz
- Un campionamento più fitto (ovvero con una frequenza di campionamento maggiore) consente di rappresentare i segnali analogici con maggiore fedeltà



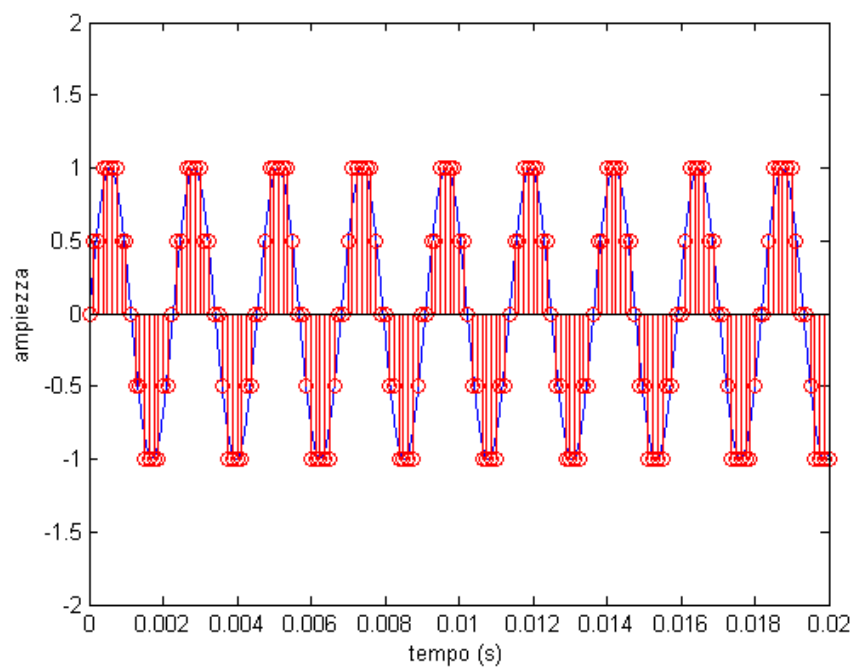
Quantizzazione

- I campioni estratti con la quantizzazione rappresentano le ampiezze con precisione arbitraria
- Per poter essere rappresentato da un calcolatore, il valore dell'ampiezza deve essere espresso tramite un numero finito di bit
- La quantizzazione suddivide l'intervallo dei valori ammissibili in 2^k bit, dove k è il numero di bit per campione.
- La figura mostra una quantizzazione a 3 bit / campione

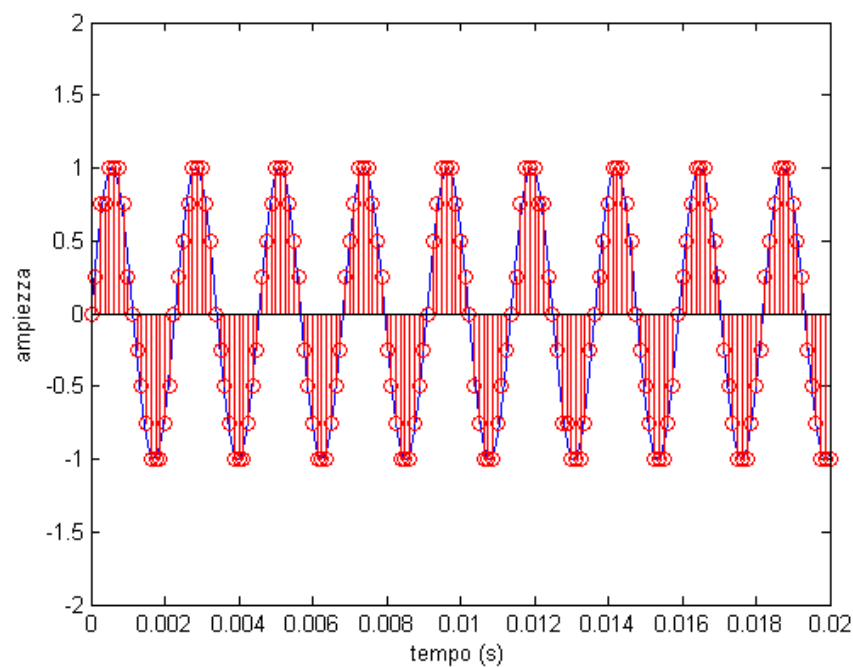




Quantizzazione



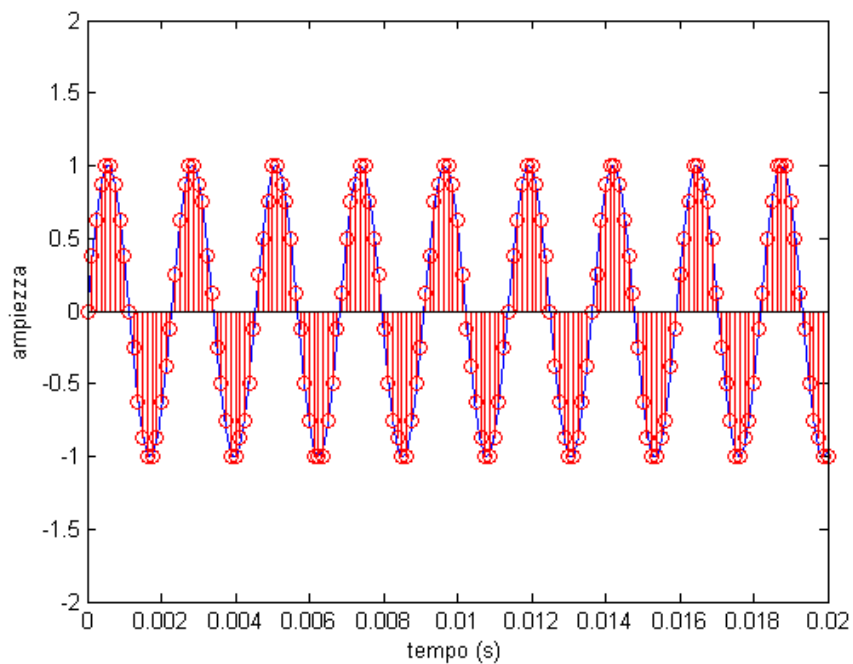
2 bit / campione



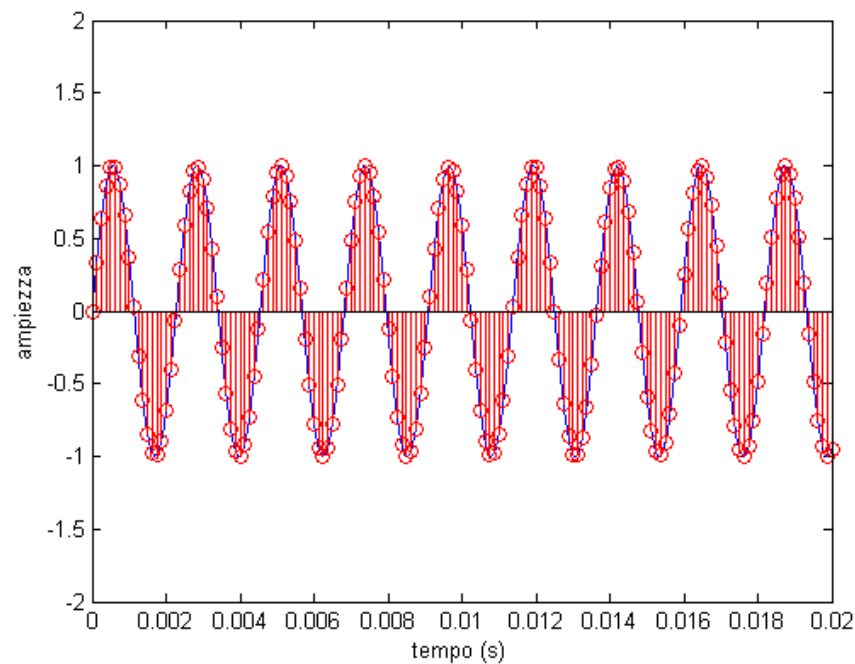
3 bit / campione



Quantizzazione



4 bit / campione



8 bit / campione



Codifica audio

- Esempio: brano musicale su CD
 - Frequenza di campionamento 44100 Hz
 - 16 bit / campione
 - 2 canali (destro e sinistro)
 - $2 \times 16 \times 44100 = 1411200$ bit / sec $\rightarrow$ 176 Kbyte / sec
 - 1 canzone
 - c.a. 5 minuti $\rightarrow 176 \text{ Kbyte/s} \times 60 \text{ sec/min} \times 5 \text{ min} \rightarrow$ c.a 52 Mbyte
 - 60 min di musica
 - $176 \times 60 \times 70 \rightarrow$ c.a. 630 Mbyte
- Un file mp3 di una canzone da 5 min sono c.a. 5 Mbyte $\rightarrow$ fattore di compressione c.a. 10:1

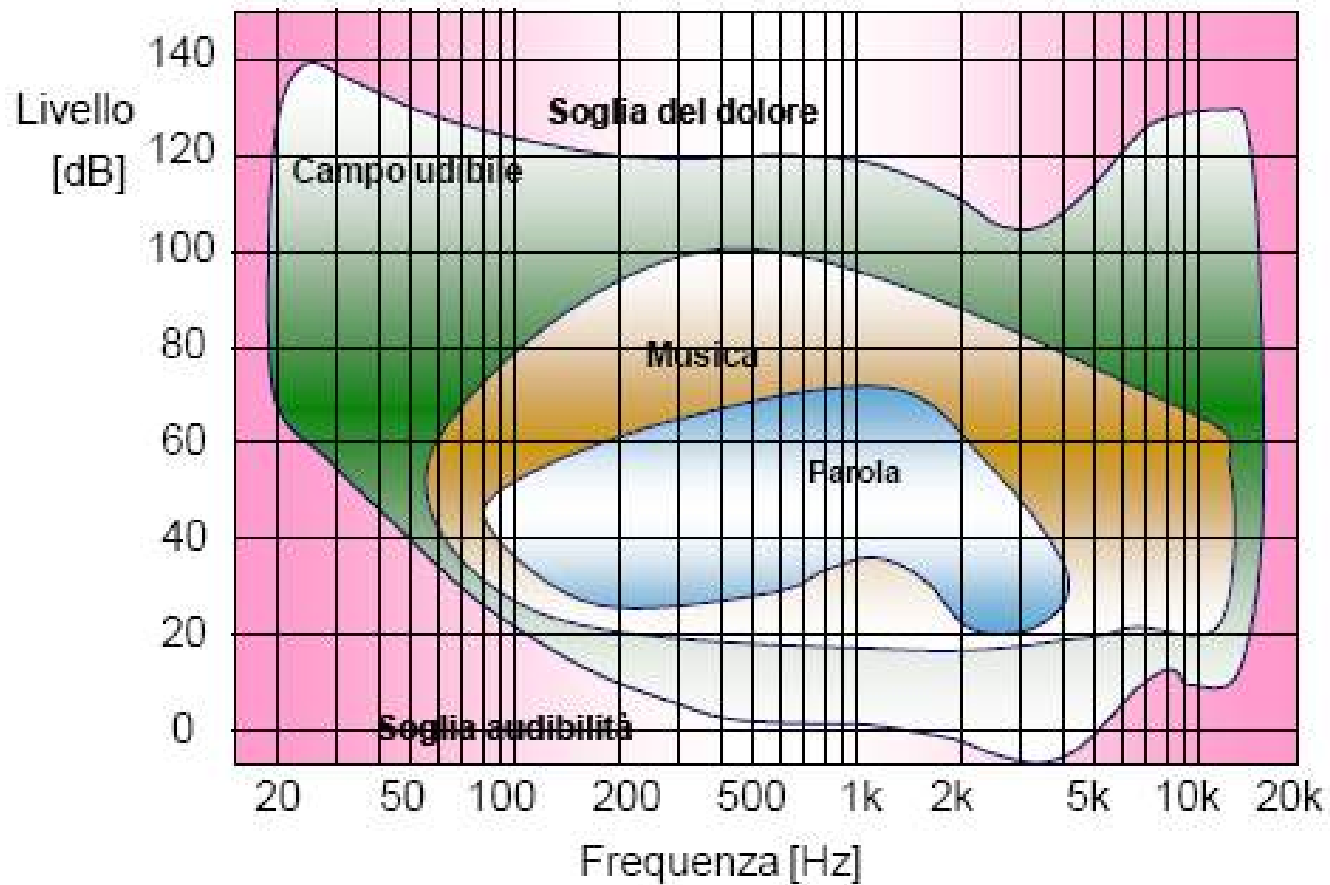


Codifica audio

- L' accuratezza della ricostruzione dipende:
 - Da quanto sono piccoli gli intervalli di campionamento
 - Da quanti bit uso per descrivere il suono in ogni campione nella fase di quantizzazione
 - Al solito... *maggiore accuratezza significa maggior quantità di memoria occupata!*
- Anche per i suoni si possono utilizzare tecniche (lossless o lossy) di compressione per migliorare l' occupazione di memoria della sequenza di campioni
- Le tecniche più efficaci (lossy) sfruttano le “debolezze” dell' orecchio umano
 - Es. MPEG-I/2 Layer 3 , detto anche MP3



Codifica audio





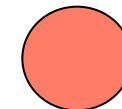
Rappresentazione di immagini

- Il calcolatore non può direttamente rappresentare in memoria informazione continua
- Le immagini devono prima essere ‘discretizzate’ ovvero trasformate in un insieme di parti distinte, che possono poi essere codificate separatamente sotto forma di numeri
- Tipologia:
 - Immagini scalari o *raster*, es. foto
 - Immagini *vettoriali*, es. un disegno geometrico

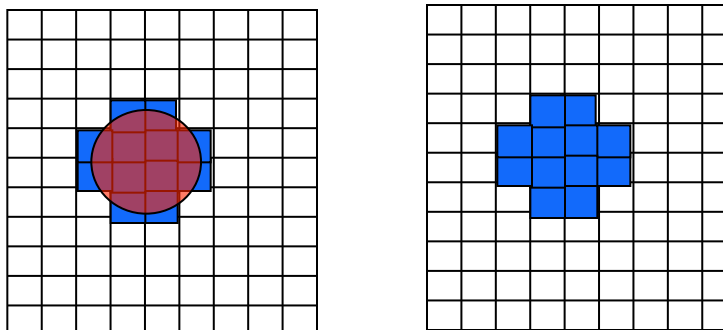


Immagini Raster

Rappresentare il disco seguente



Si sovrappone al disco una griglia



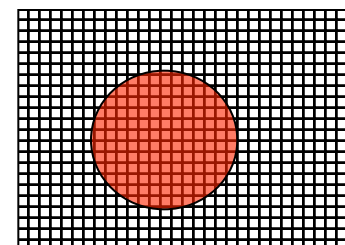
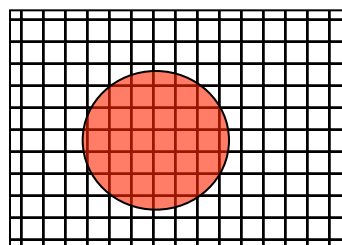
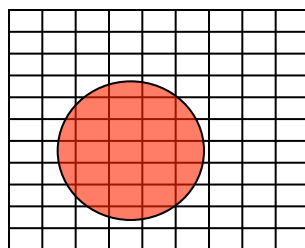
Si identificano i “quadratini” (in azzurro) che
“discretizzano” il disco

Ogni quadratino si dice *Pixel* (da picture element)



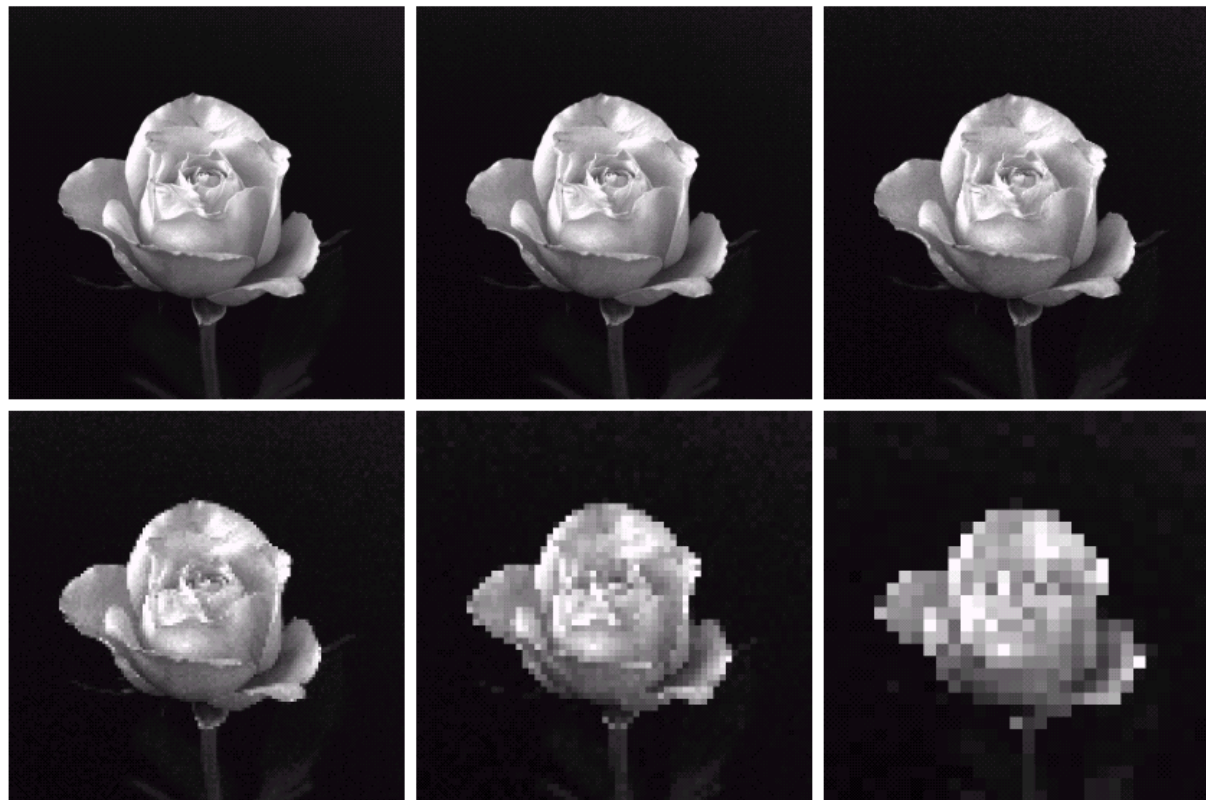
Risoluzione

- Chiamiamo *risoluzione* dell'immagine la dimensione della griglia utilizzata per discretizzare l'immagine
 - es. 640x480
- Aumentando la risoluzione (ovvero il numero dei pixel) e quindi diminuendo la dimensione del singolo pixel, la rappresentazione approssima meglio l'immagine originaria





Effetti della riduzione di risoluzione



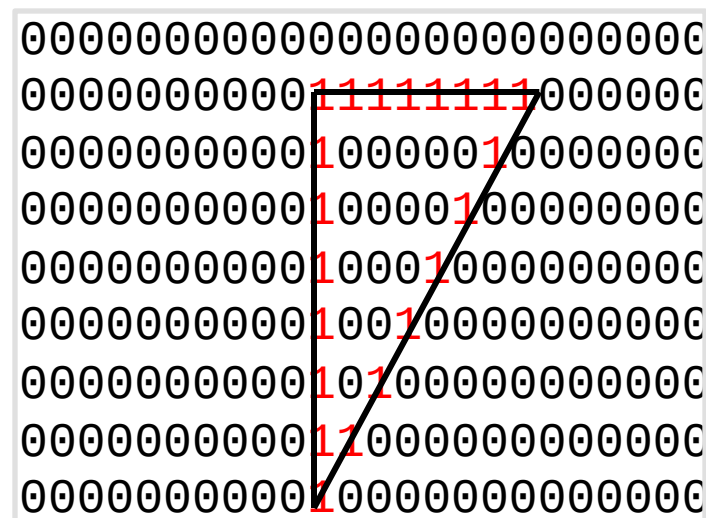
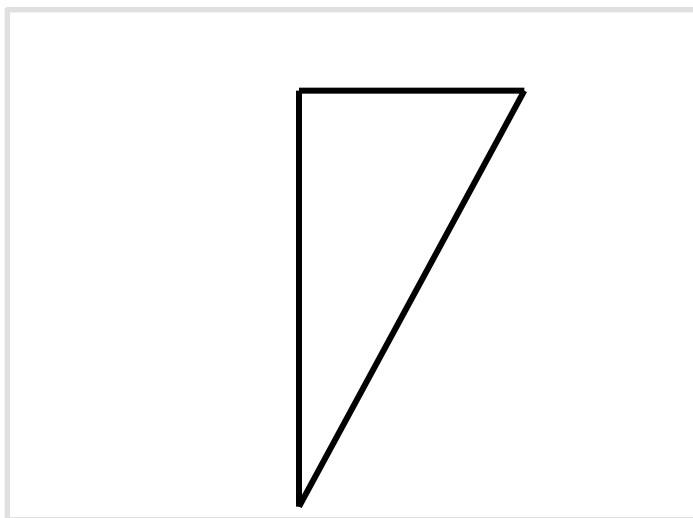


Quantizzazione

- Dopo aver discretizzato l'immagine occorre rappresentare ogni pixel con un numero
- Tale numero dovrà rappresentare il colore associato al pixel, usando un certo range: si parla di *quantizzazione*
- Nel caso di immagini in bianco e nero senza sfumature sono sufficienti due soli bit per ogni pixel:
 - 0 per rappresentare i pixel più bianchi
 - 1 per rappresentare i pixel più neri

Immagini b/n: Un esempio

Ogni pixel è rappresentato da un bit





Immagini in scala di grigi / colori

- Immagini in bianco e nero con sfumature
 - Si usa la rappresentazione in ‘toni di grigio’ : un byte per pixel, con 256 gradazioni di grigio per ogni pixel, o più byte per pixel, per avere più gradazioni possibili
- Immagini a colori
 - Si usa la rappresentazione del colore secondo il modello RGB (red, green, blu): sintesi additiva dei tre colori primari red, green, blu, ognuno con la propria codifica
 - *Profondità cromatica*: numero di bit per ogni pixel (di norma 24, ovvero un byte per ogni colore)

RED	100	Codifica : 01000100
GREEN	200	Codifica : 11001000
BLU	200	Codifica : 11001000





Effetti della quantizzazione

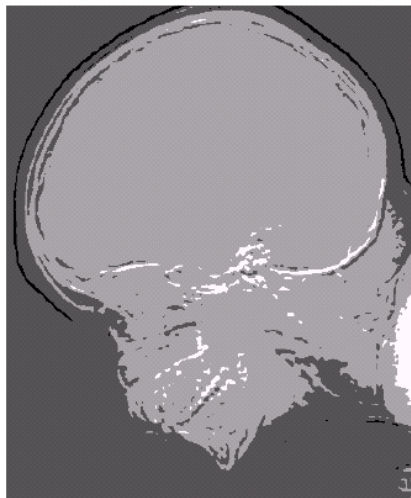
4 bpp



3 bpp

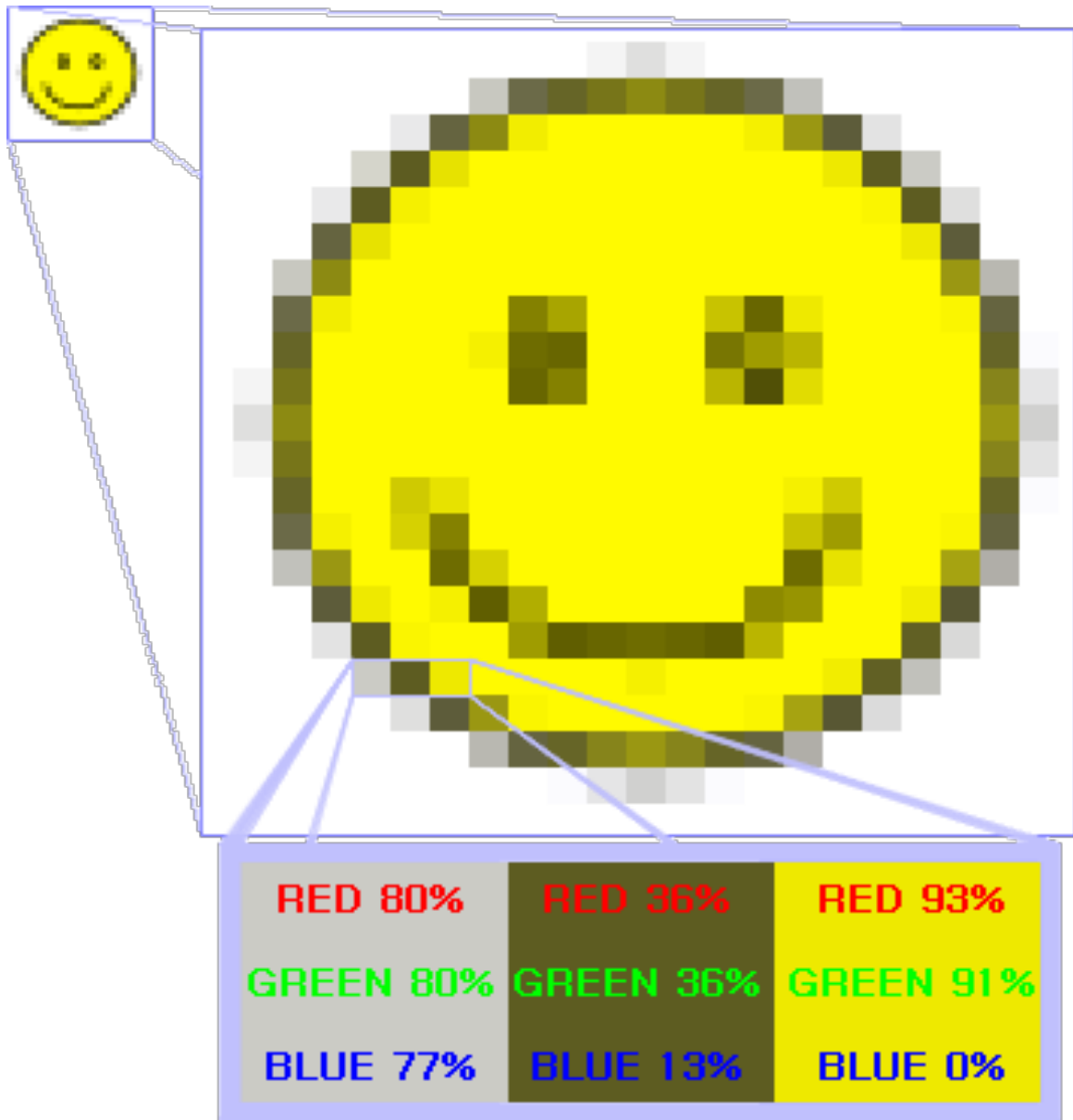


2 bpp



1 bpp







Codifica di immagini

- La risoluzione e la profondità cromatica determinano la dimensione di memoria necessaria a memorizzare l'immagine
 - Esempio: $1024 \times 768 \text{ pixel} \times 256 \text{ toni di grigio} / \text{pixel} =$
 $1024 \times 768 \text{ pixel} \times 8 \text{ bit} / \text{pixel} = 768 \text{ Kbyte}$
- Se l'immagine è a colori
 - Esempio $1024 \times 768 \text{ pixel} \times 3 \text{ componenti} / \text{pixel} \times 256$
 $\text{toni} / \text{componente} =$
 $1024 \times 768 \text{ pixel} \times 3 \text{ componenti} / \text{pixel} \times 8$
 $\text{bit} / \text{pixel} = 2304 \text{ Kbyte}$



Codifica di immagini

- In fase di codifica esiste la necessità di adottare tecniche di compressione per ottimizzare:
 - Occupazione di spazio di memoria
 - Velocità di trasmissione attraverso la rete
- Tecniche di compressione senza perdita d informazione (*lossless*)
 - Reversibili
 - Ad hoc per le immagini → ad es. PNG
 - Applicabili a qualsiasi tipo di dato → ad es. codifiche basate sulla frequenza statistica dei dati: i dati più frequenti vengono codificati con sequenze di bit più brevi (ZIP)
- Algoritmi con perdita di informazione (*lossy*)
 - Generalmente sono specifici di un certo campo e sfruttano le caratteristiche degli oggetti da rappresentare per ‘buttare via’ informazione poco importanti
 - Nel caso di immagini gli algoritmi usati nei formati GIF e JPEG sfruttano la caratteristica dell’occhio umano di essere poco sensibile a lievi cambiamenti di colore in punti contigui, e quindi eliminano questi lievi cambiamenti “appiattendo” il colore dell’immagine
 - Generalmente è possibile specificare quanto siamo disposti a perdere attraverso alcuni parametri

Principali formati di compressione per immagini raster

- **TIFF** (*Tagged Image File Format*): uso di tag (etichette) descrittivi, 24 bit/pixel, compressione senza perdita
- **GIF** (*Graphics Interchange Format*, CompuServe): più immagini nello stesso file, compressione senza perdita
- **PNG** (*Portable Network Graphics*): compressione lossless, studiato per sostituire GIF (coperto da brevetti). Supporta solo grayscale e RGB. Studiato per trasmissione di immagini su Web
- **BMP** (*BitMaP*, Microsoft e IBM):
1, 4, 8, 24 bit/pixel,
compressione senza perdita (RLE)
- **JFIF** (*Jpeg File Interchange Format*):
compressione JPEG
(meglio noto come “formato JPEG”)





Fattore di Compressione

- Esempio:
 - Fotocamera a 8 Mpixel $\rightarrow$ c.a. 8.000.000 di pixel
- Occupazione immagine non compressa a colori
 - 8.000.000 pixel $\times$
 - 3 componenti / pixel $\times$
 - 1 byte / componente $=$

 - 24.000.000 byte $\rightarrow$ c.a. 24 Mbyte
- Salvo l'immagine in formato JPEG e ottengo un file mypicture.jpg di dimensione 2.4 Mbyte
- Fattore di compressione $\rightarrow$ dimensione originale / dimensione file compresso
 - 24 Mbyte / 2.4 Mbyte = 10:1



Esempio: Compressione jpeg



Compressione = 13.6:1



Compressione = 37:1



Compressione = 78:1



Compressione = 96:1



Immagine originale 8 bpp

bpp $\rightarrow$ bits / pixel





Immagine originale (dettaglio)





Compressione JPEG 0.5 bpp

- Fattore di compressione
 - $8 \text{ bpp} / 0.5 \text{ bpp} = 16$





Compressione JPEG 0.5 bpp (dettaglio)





Compressione JPEG 0.35 bpp

- Fattore di compressione
 - $8 \text{ bpp} / 0.35 \text{ bpp} = 22.8$





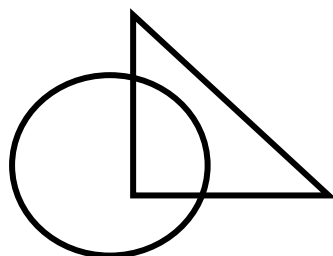
Compressione JPEG 0.35 bpp





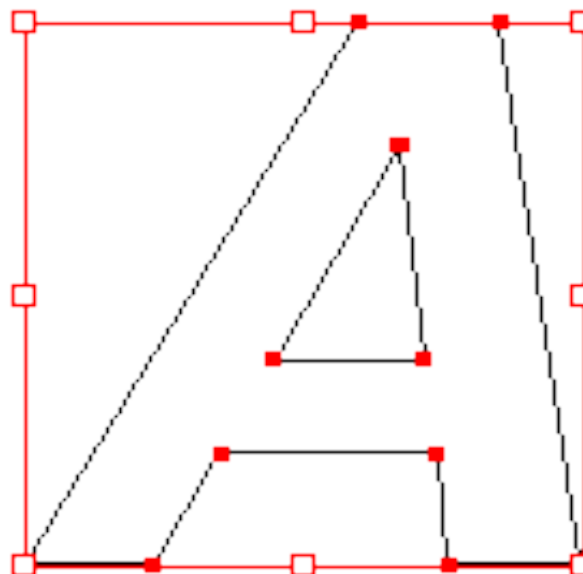
Immagini vettoriali (1/2)

- Definiscono gli oggetti che compongono l'immagine mediante equazioni matematiche



circle
polyline

.....



- Basate su una rappresentazione geometrica delle immagini: usano relazioni matematiche tra punti e linee per descrivere una immagine
- Nessuna perdita di dettaglio ingrandendo o rimpicciolendo l'immagine

Immagini vettoriali (2/2)

Le immagini vettoriali sono composte da OGGETTI.

Tutti gli oggetti sono costruiti a partire da una serie di primitive quali:

- Punti
- Linee
- Rettangoli
- Ellissi



La grafica vettoriale è l'uso di primitive geometriche quali punti, linee, curve e forme o poligono (s), che sono tutte basate su equazioni matematiche, per rappresentare le immagini nel computer.



Immagini vettoriali: pro e contro

Vantaggi:

- Fatta di oggetti geometrici che formano gli oggetti
 - Line ($x1, y1, x2, y2$)
 - Circle($x, y, radius$)
- Può essere scalata a qualsiasi dimensione senza perdere qualità
- Indipendente dalla risoluzione
- Occupano pochissimo spazio in termini di memoria (dell'ordine dei kilobytes)



Immagini vettoriali: pro e contro

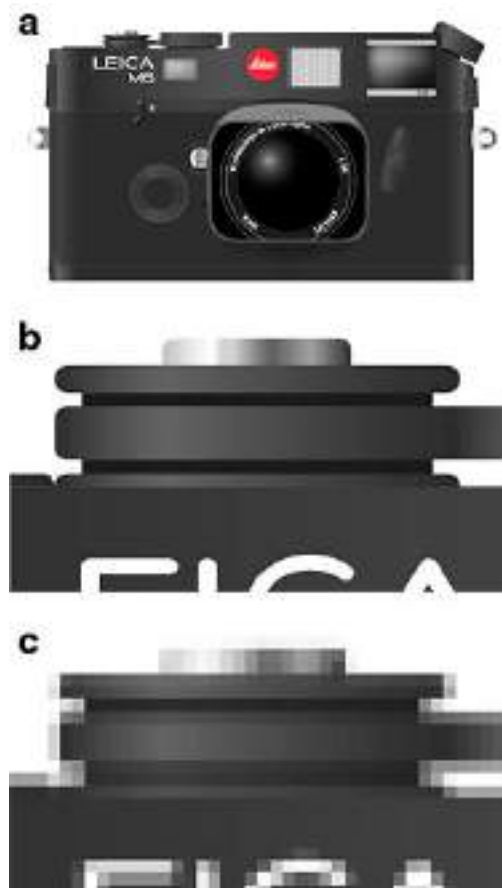
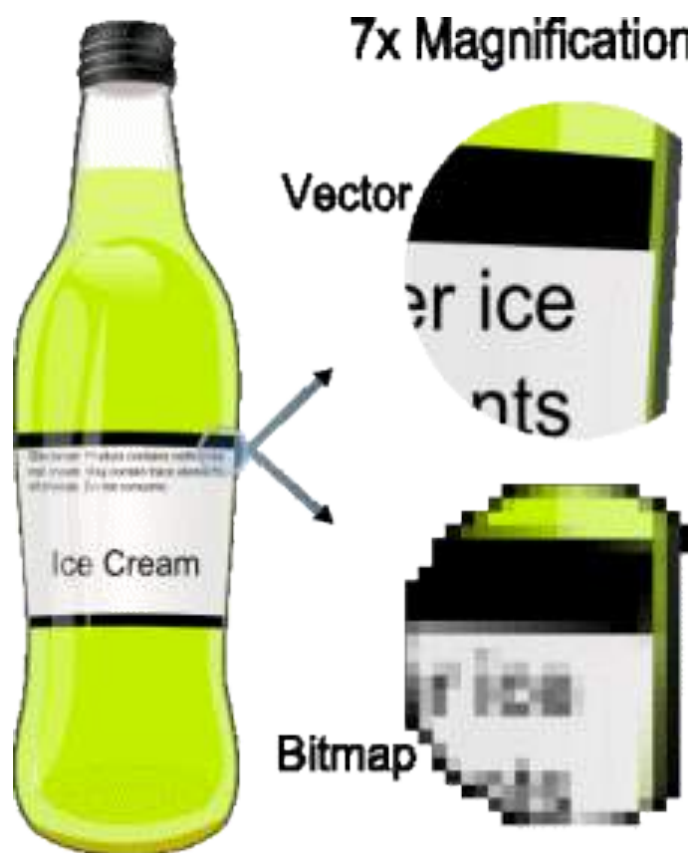
Svantaggi:

- Le immagini vettoriali sono per loro natura generate all'interno della macchina
- Non è il miglior formato per immagini di tipo fotografico con toni continui, luci, ombre, miscele di colore.

Es: E' possibile rappresentare in maniera schematica un oggetto come un albero e quindi renderne l'idea, ma è impossibile dettagliarne ogni particolare come ad esempio le foglie, le striature sul tronco ecc.

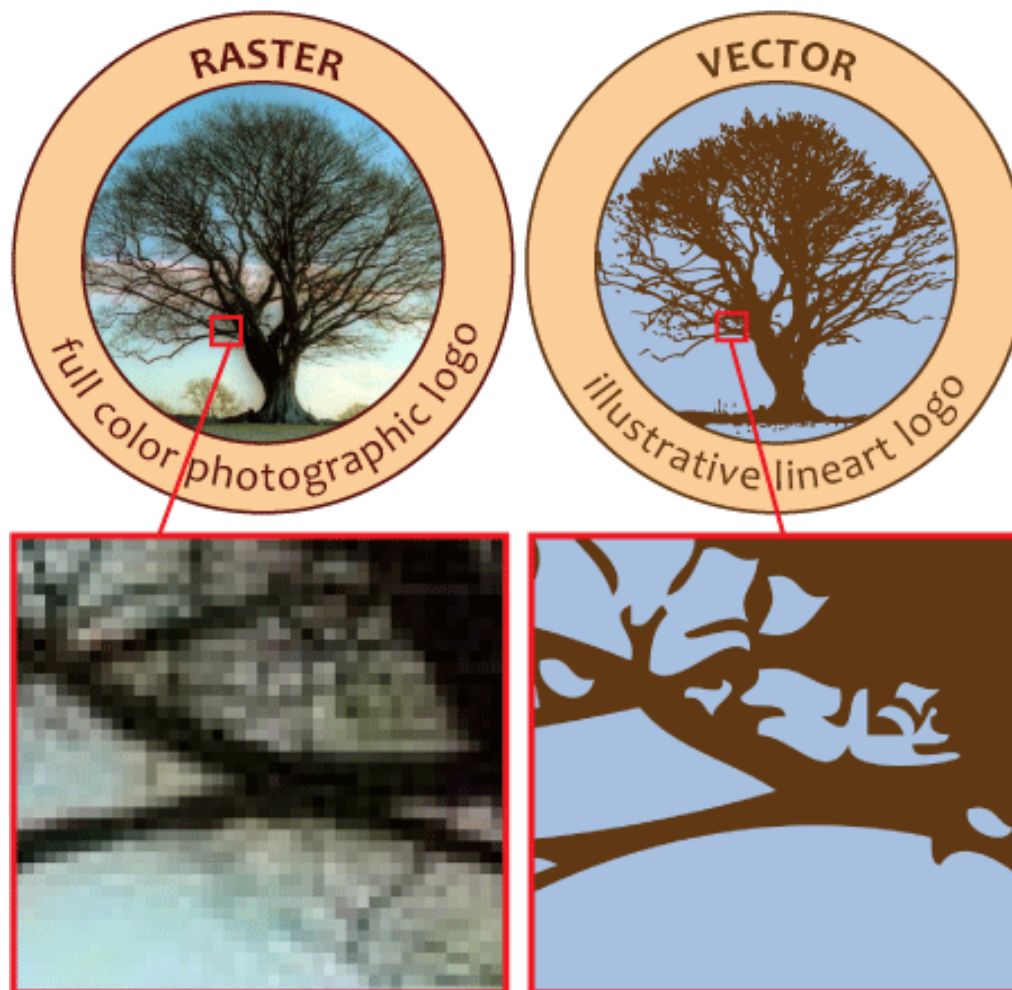


Immagini Vettoriali / Raster: esempi





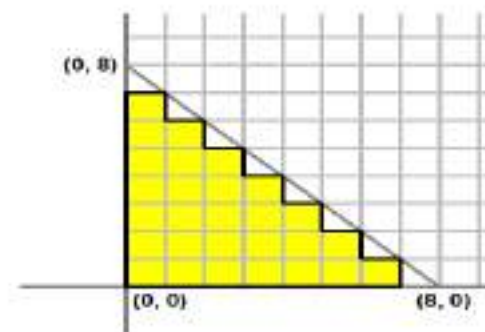
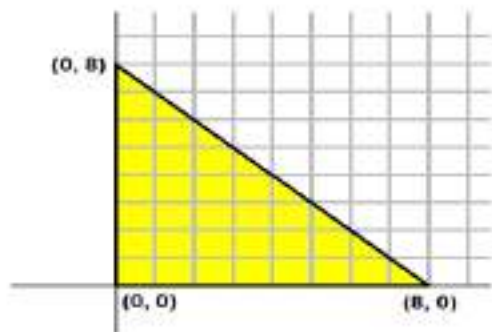
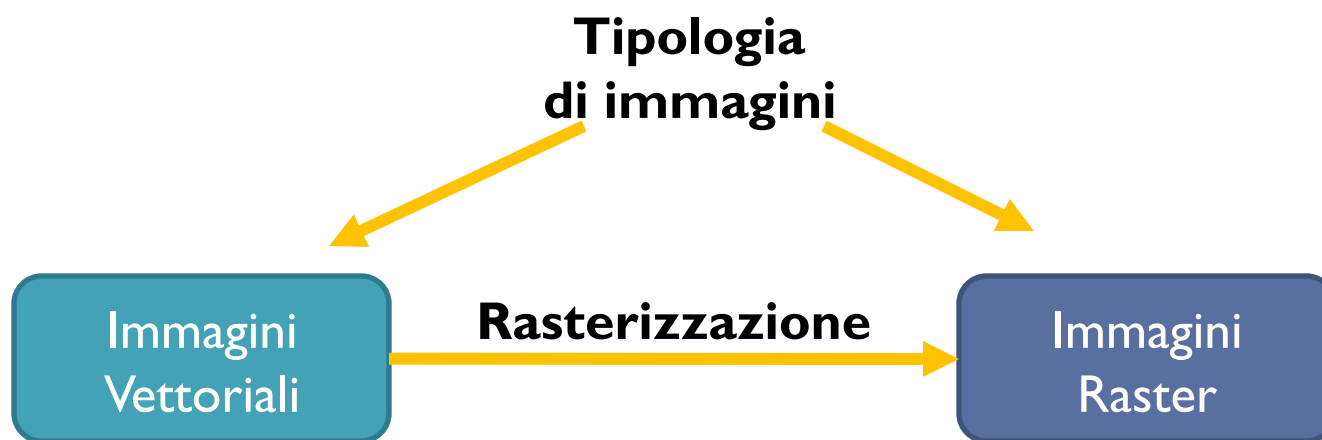
Immagini Vettoriali / Raster: esempi





Rasterizzazione

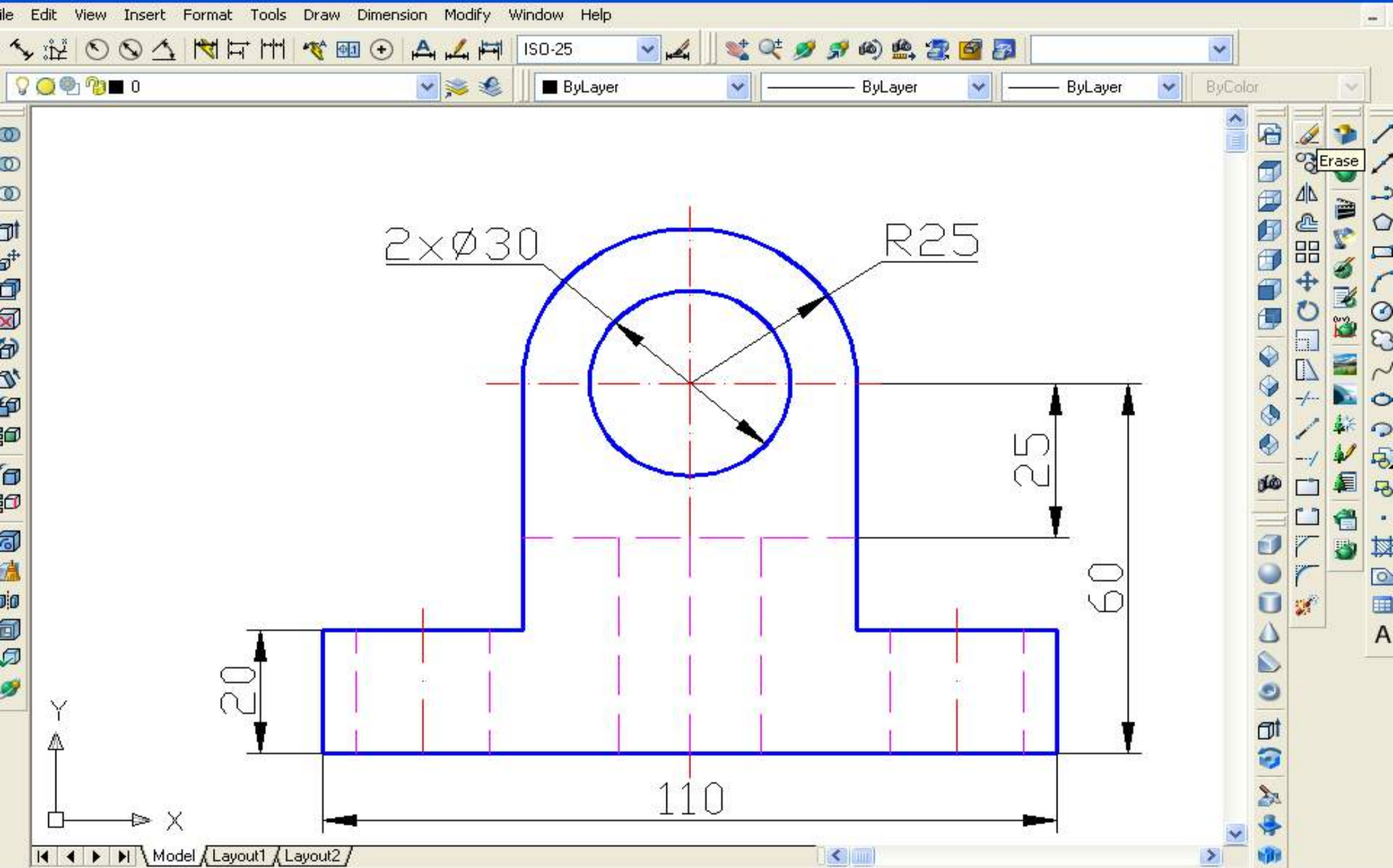
Convertire un'immagine vettoriale in un'immagine raster





AutoCAD

- Software Computer Aided Design (CAD) per la creazione di disegni bi/tridimensionali in ambito ingegneristico, architettonico, meccanico, etc.
- L'immagine prodotta è di tipo vettoriale (composta da entità grafiche definite come oggetti matematico/geometrici)
→ scalata a qualsiasi dimensione senza perdere qualità
- Formati di AutoCAD:
 - DWG (drawing); formato nativo di Autocad
 - DXF (drawing exchange format), formato standard di interscambio di dati CAD tra diverse applicazioni.



Command:
Erases objects from a drawing: ERASE



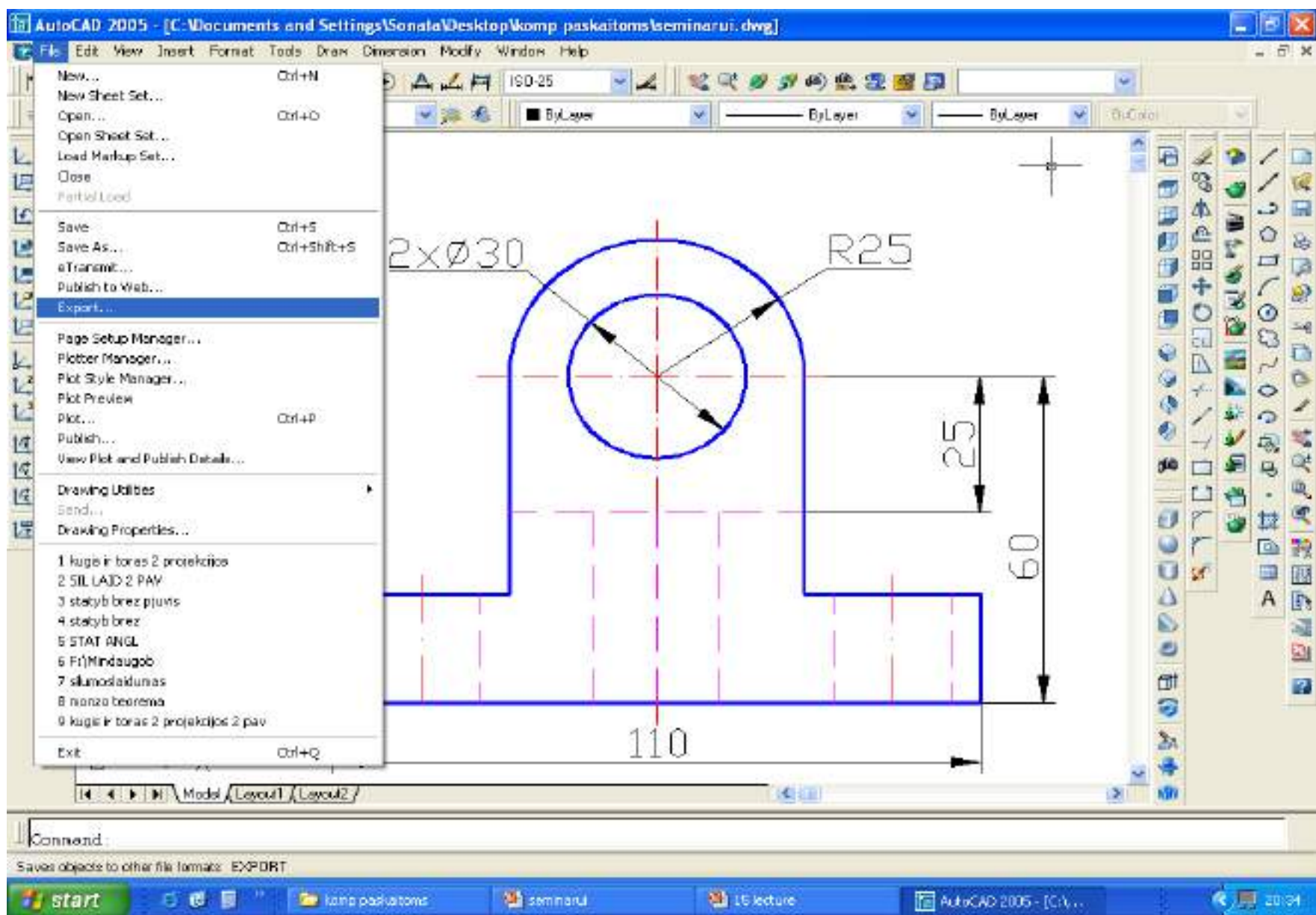
AutoCAD DWG

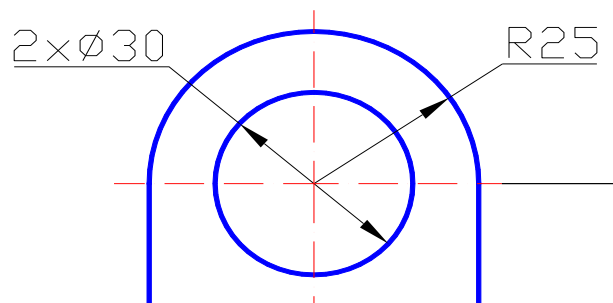
- DWG è l'abbreviazione di drawing, ovvero "disegno"
- Formato nativo per i file di tipo CAD, sviluppato da Autodesk come formato di memorizzazione del disegno in AutoCAD (ed altri SW della stessa piattaforma)
- DWG è un file binario suddiviso in varie sezioni e con una organizzazione complessa dei dati.:
 - Si accede al file tramite AutoCAD e non direttamente
- Il DWG non è uno standard in senso stretto ma è standard di fatto:
 - Autodesk non ha mai rilasciato le specifiche
 - uno strumento poco adatto allo scambio di dati tra CAD diversi.



AutoCAD DXF

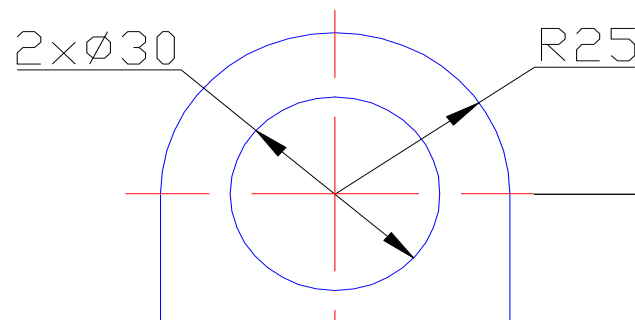
- DXF (Drawing Interchange Format, o Drawing Exchange Format)
 - Formato sviluppato da [Autodesk](#) per scambiare file CAD tra [AutoCAD](#) e programmi di terze parti.
 - Introdotto nel 1982 insieme con AutoCAD 1.0
- Consente una rappresentazione esatta dei dati del formato proprietario [DWG](#), per il quale Autodesk non ha mai rilasciato le specifiche.



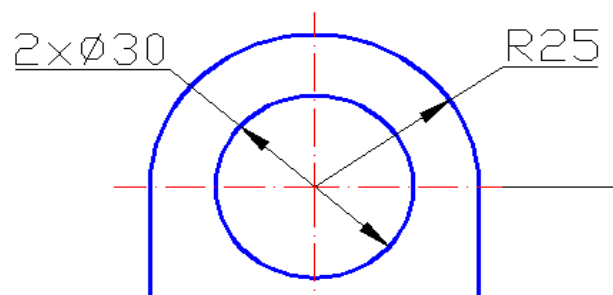


WMF format

Formati
Vettoriali

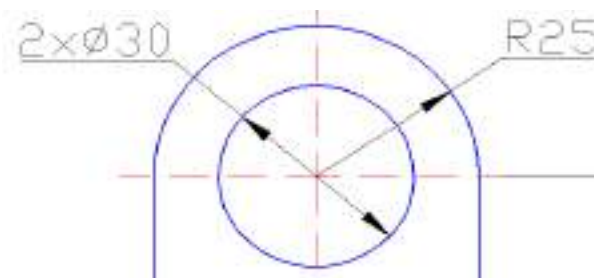


EPS format

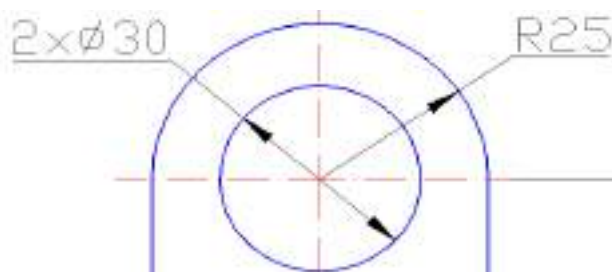


BMP format

Formati
Raster



JPG format



PNG format



Formati vettoriali/bitmap (misti)

- **PostScript** (Adobe): misto bitmap-vettoriale, compressione con e senza perdita
 - È un linguaggio di descrizione di pagina utilizzato principalmente nel mondo dell' editoria elettronica e del desktop publishing
- **PDF** (*Portable Document Format*): derivato da PostScript
 - Utilizzato per rappresentare documenti 2D in modo indipendente dal dispositivo di rendering e della risoluzione dello stesso
 - Ultima versione PDF 1.7 (2006)
 - Contiene
 - Un sottoinsieme del linguaggio PS
 - Un sistema per l' embedding/replacement dei font
 - Un sistema per unire in modo strutturato contenuti di tipo differente (testo, immagini, font, grafica vettoriale)



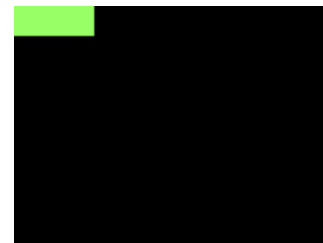
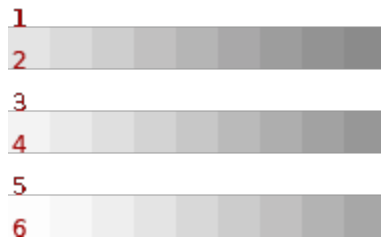
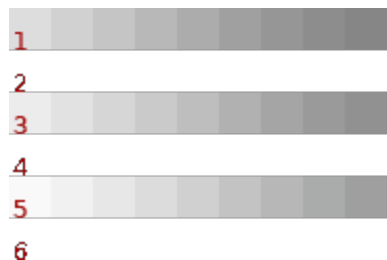
Immagini in movimento: i video

- Video: successione di immagini fisse (o frame) trasmesse con velocità sufficientemente elevata
- Il movimento è rappresentato già in modo discreto nei media: con un numero abbastanza alto di fotogrammi fissi (15-30 al secondo) l'occhio umano percepisce il movimento come un continuo



Video interlacciato vs. progressivo

- Tradizionalmente, il segnale video di tipo televisivo utilizza un formato **interlacciato**
- Prima vengono generate le righe pari, poi quelle dispari



- Vengono generati 50 “semiquadri” al secondo
- Nel formato **progressivo**, ogni frame è costituito sia dalle righe pari che da quelle dispari



Un esempio – video interallacciato

- Interallacciato (ad. es TV)
 - 720 punti / linea
 - 576 linee (288 linee pari, 288 linee dispari)
 - 50 semiquadri al secondo
 - 3 byte / pixel (RGB)
 - $720 \times 288 \times 50 \times 3 = 31.104.000$ byte / sec
 - c.a. 31 Mbyte/sec → c.a. 250 Mbps
 - 2h di film occupano (se non compressi)
 - $2 \times 60 \times 60 \times 31 = 223.200$ Mbyte → c.a. 223 Gbyte
 - Un DVD ha una capacità di c.a. 4.5 Gb → avrei bisogno di un fattore di compressione pari a $223 / 4.5 = 50$



Un esempio - video progressivo

- Progressivo (ad. es cellulare)
 - 320x240 pixel
 - 15 frame / secondo
 - 3 byte / pixel (RGB)
 - $320 \times 240 \times 15 \times 3 = 3.456.000$ byte / sec
→ c.a. 3.5 Mbyte/sec → c.a. 28 Mbyte / sec
 - 1 minuto di video registrato occupa (se non compresso)
 - $60 \times 3.5 = 210$ Mbyte



Codifica video

- Potrei in principio codificare separatamente ogni fotogramma come immagine fissa
- E' la tecnica usata da molte fotocamere compatte, che salvano i filmati ripresi in formato Motion-JPEG (ogni frame è compressa con JPEG)
- Si è in grado di ottenere rapporti di compressione dell'ordine 10-20 senza eccessiva perdita di qualità
 - Per mettere un film su DVD abbiamo bisogno di un rapporto di c.a. 50...



Codifica video

- Per ottenere rapporti di compressione più alti è necessario sfruttare la ridondanza temporale
- Frame consecutivi in una sequenza di immagini sono simili l'uno all'altro
- Codifico solo le differenze tra i frame successivi
- E' possibile ottenere rapporti di compressione di 50:1 fino a anche a 100:1 senza un degrado eccessivo della qualità



Immagine originale I

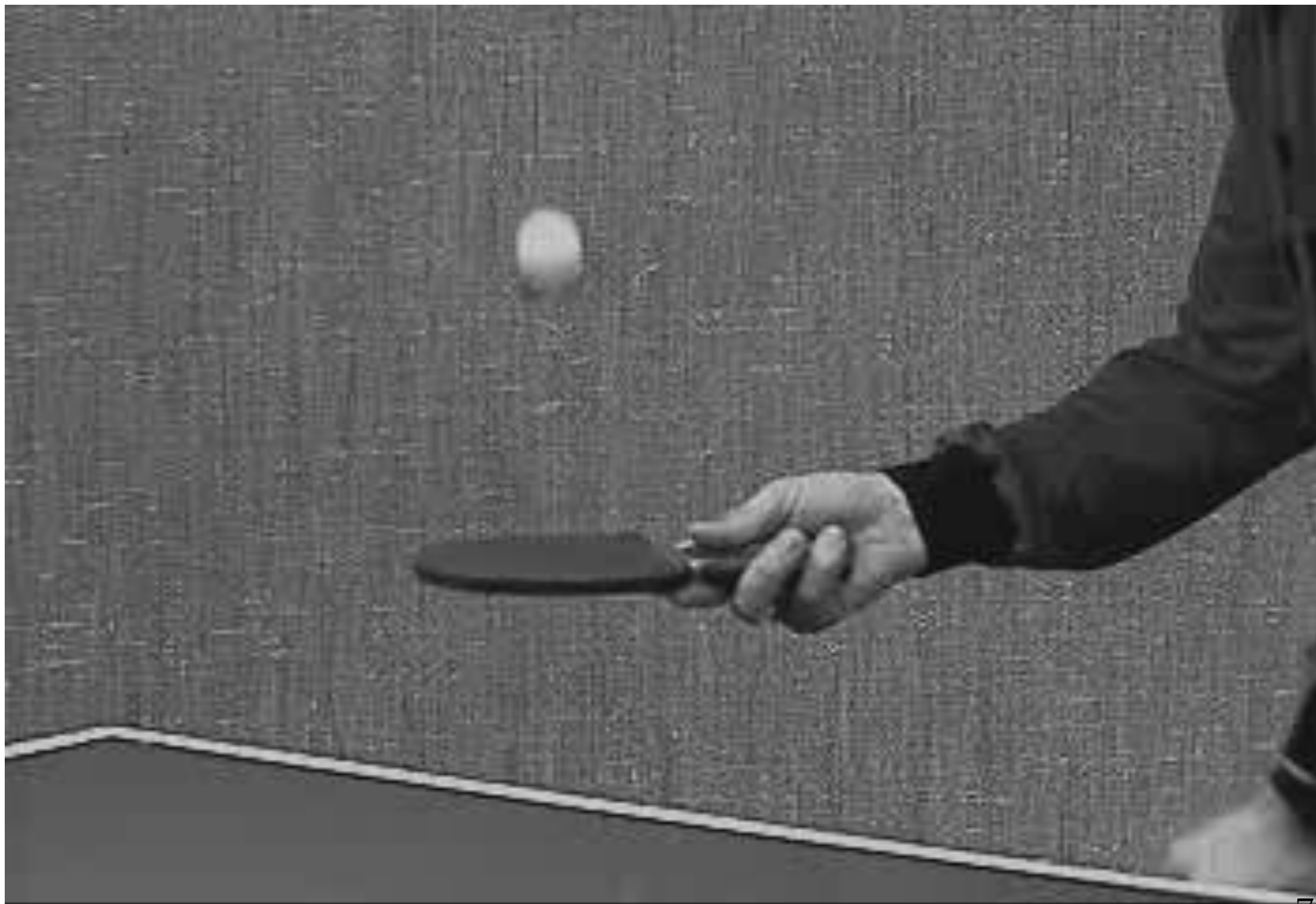
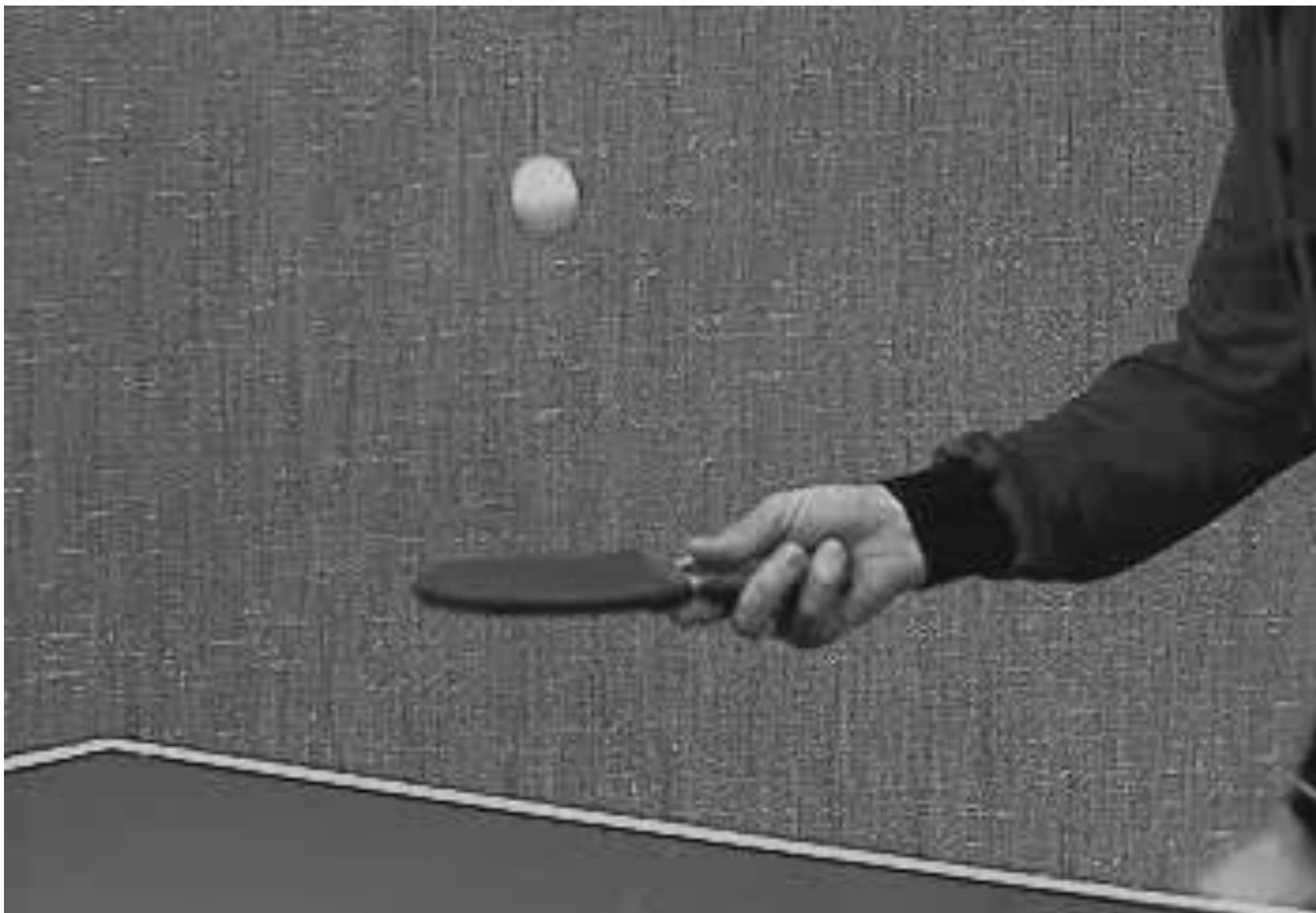


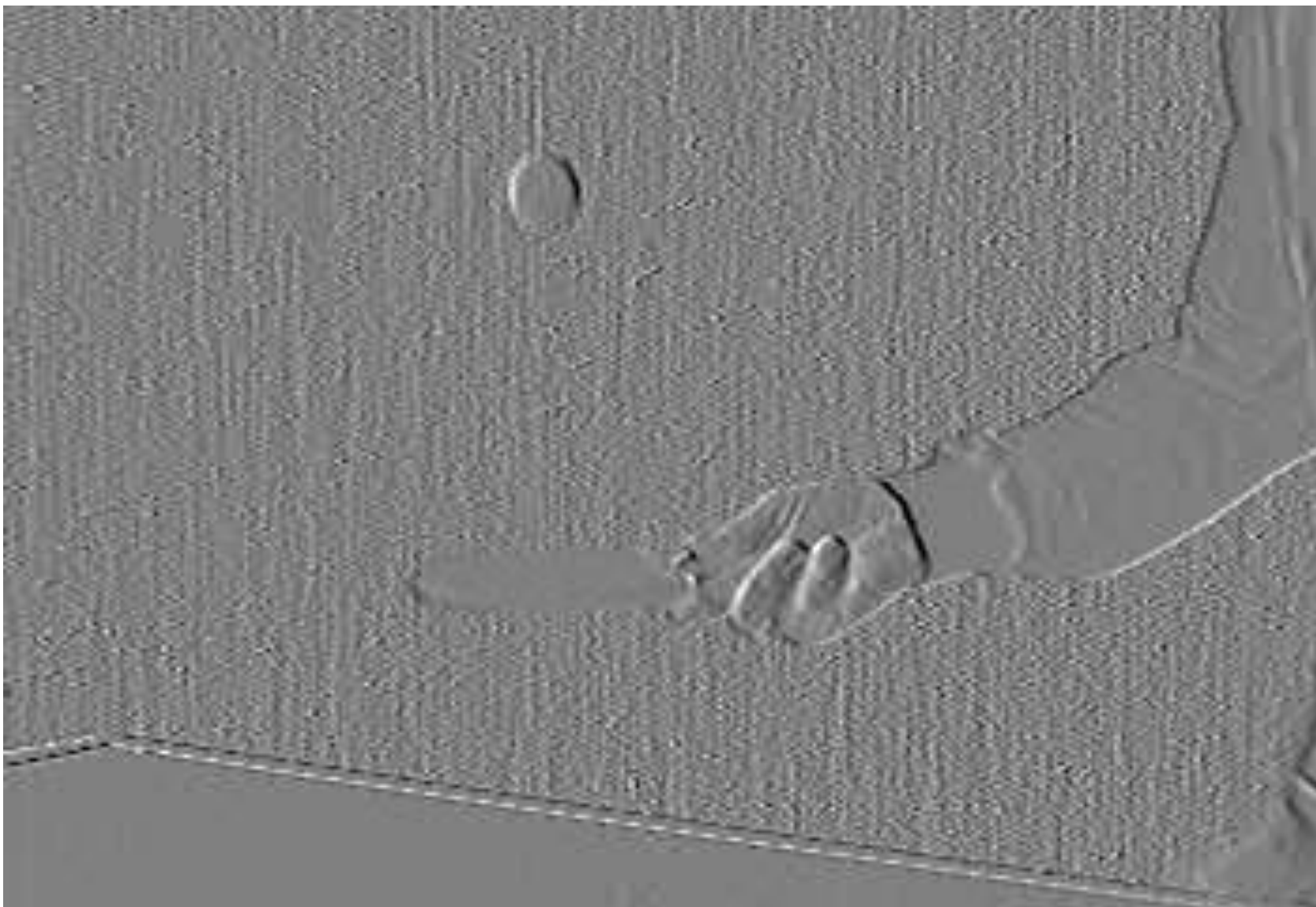


Immagine originale 2



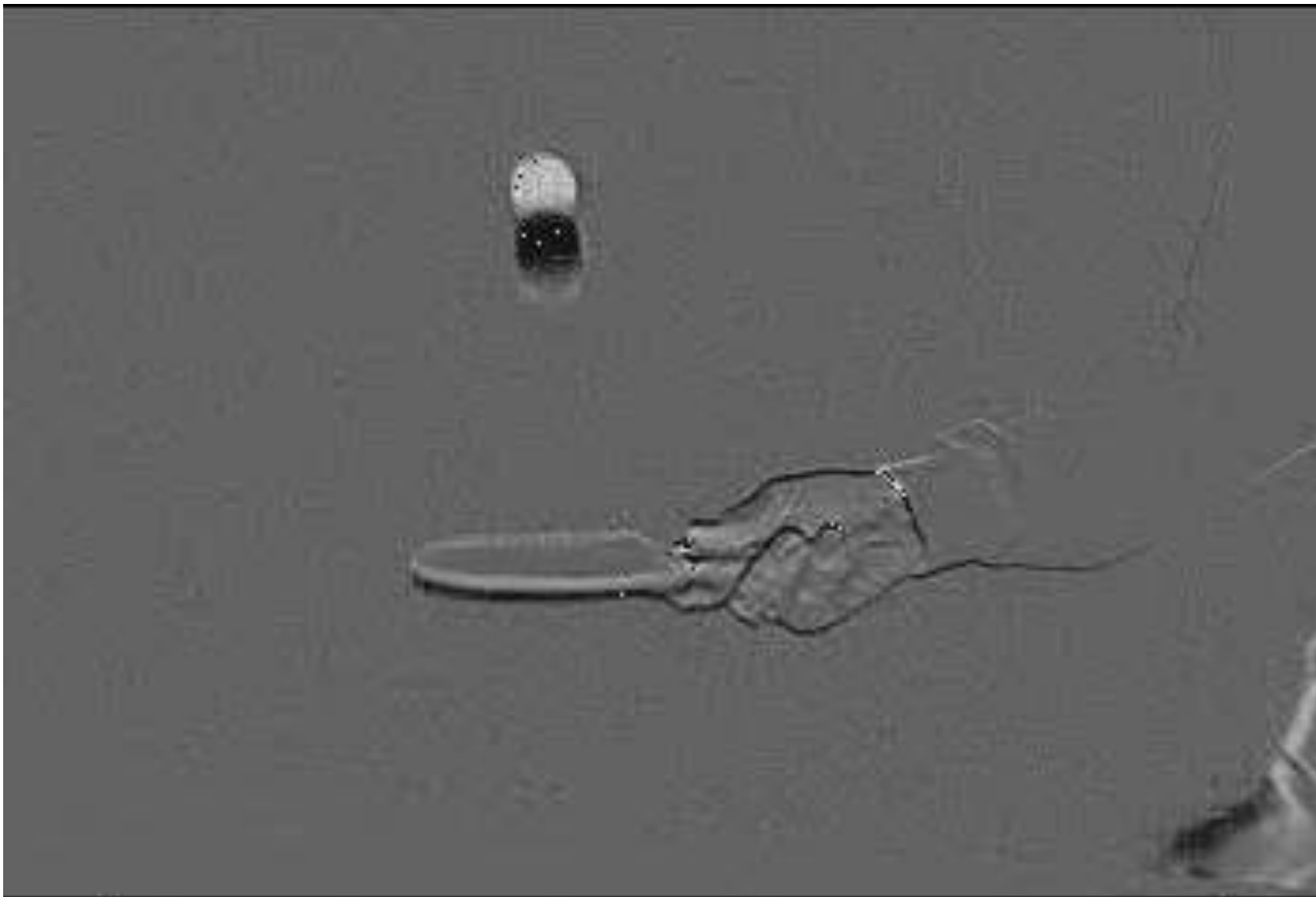


Differenza con pixel di destra



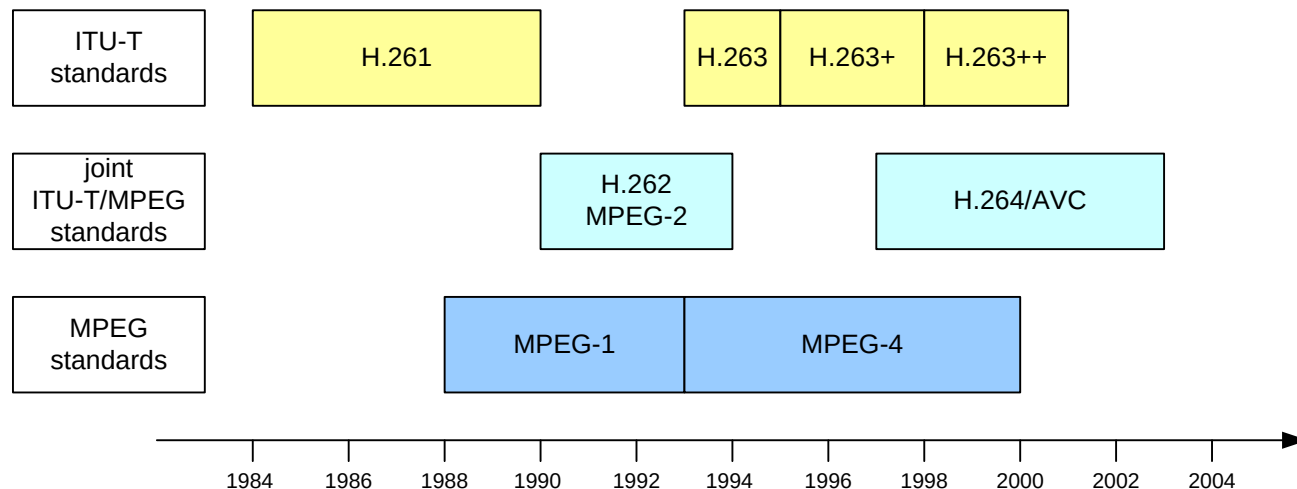


Differenza tra originali





Standard di codifica video



- **MPEG** (Moving Picture Experts Group):
 - MPEG-1 → Video CD
 - MPEG-2 → TV digitale, DVD
 - MPEG-4 → DivX,
 - H.264/AVC → HD-DVD, Blue-ray DVD
 - Definisce anche la codifica dei dati audio (vedi slide seguenti) e la sincronizzazione con i dati video
- **ITU-T** (International Telecommunication Union - Telecommunications Standardization Sector)
 - H.263 → video-telefonia
- **AVI** (Audio Video, Microsoft)
 - Formato “contenitore”. Può contenere audio/video compressi con formati diversi