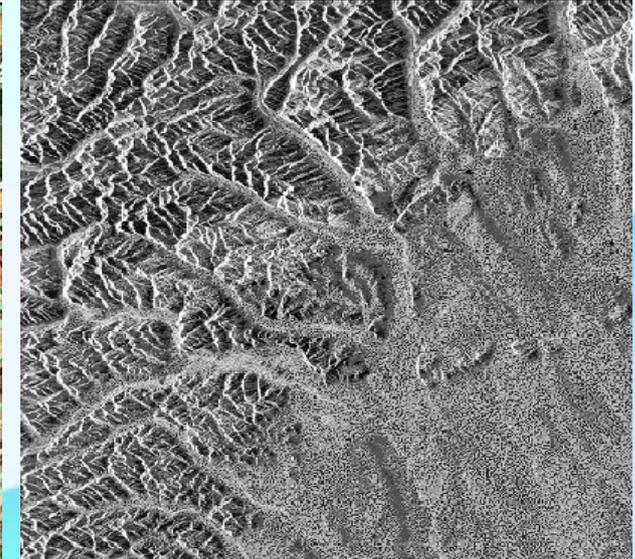
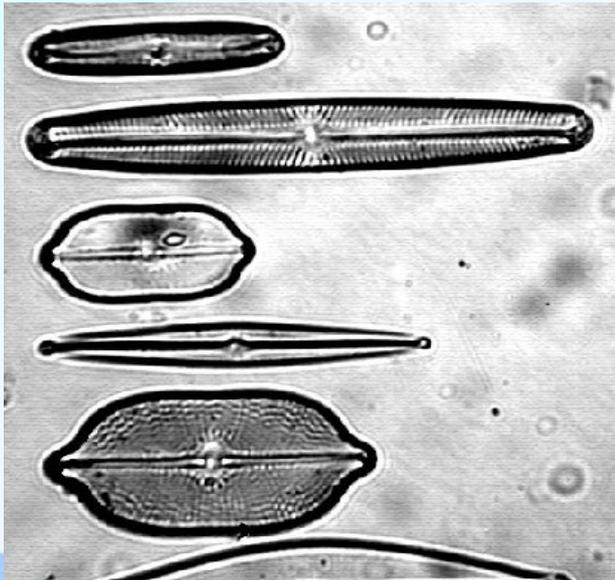
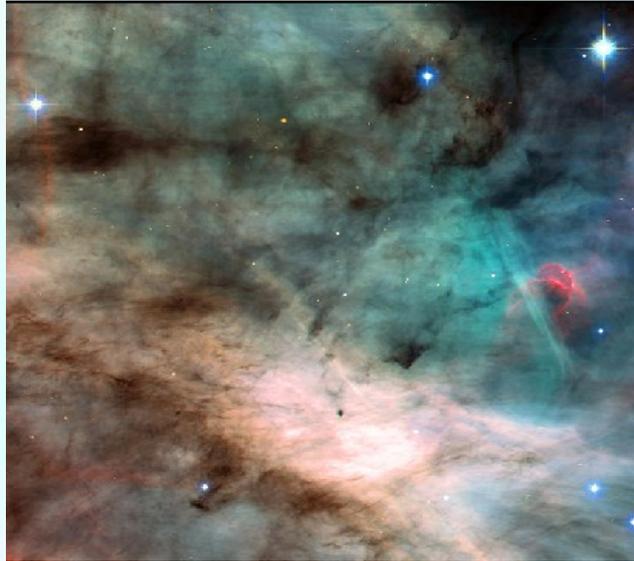


# Immagini Digitali (parte 4)



# Formati di Immagini

Come le immagini sono archiviate

# Data storage and transmission

## ➤ Problema

1) Scrivere i dati su memoria o comunicare i dati tramite canale di trasmissione (rete)

2) Rileggere i dati e ricostruire l'immagine per

1) Visualizzazione

2) Analisi

# Formati di Immagini

- Obiettivo dei diversi formati
  - Interscambio
    - Differenti applicativi
    - Differenti sistemi operativi
    - Differenti architetture hardware (endianness)
  - Storraggio
    - Archiviazione
    - Estensibilità
    - Compatibilità
  - Diffusione
    - Internet
  - Metadati, informazioni accessorie

# Rappresentazione ASCII



$F(x, y)$



148	123	52	107	123	162	172	123	64	89	...
147	130	92	95	98	130	171	155	169	163	...
141	118	121	148	117	107	144	137	136	134	...
82	106	93	172	149	131	138	114	113	129	...
57	101	72	54	109	111	104	135	106	125	...
138	135	114	82	121	110	34	76	101	111	...
138	102	128	159	168	147	116	129	124	117	...
113	89	89	109	106	126	114	150	164	145	...
120	121	123	87	85	70	119	64	79	127	...
145	141	143	134	111	124	117	113	64	112	...
:	:	:	:	:	:	:	:	:	:	:

$I(u, v)$

# Soluzione 1: formato 'human readable'

·Conversione binario -> ASCII (<https://www.asciitable.com/>) del valore del 1° pixel in alto

- Il contenuto binario del pixel  $10010100_2$  deve essere convertito nella sequenza di caratteri che rappresentano il numero  $148_{10}$
- Questo comporta la conversione nella sequenza di 3 byte contenenti nell'ordine:
  - '1' (rappresentato dal codice ASCII  $00110001_2$ )
  - '4' (rappresentato dal codice ASCII  $00110100_2$ )
  - '8' (rappresentato dal codice ASCII  $00111000_2$ )
- Questa conversione andrebbe eseguita per ogni pixel sia in scrittura che in lettura eseguendo la conversione inversa
  - Totalmente impraticabile:
    - Per rappresentare 1 byte ne dobbiamo usare 3
    - Inaccettabili costi computazionali nell'eseguire le operazioni di conversione

# Soluzione 2: Lettura/Scrittura dei dati binari

## ➤ Soluzione elementare:

➤ Il contenuto della matrice dell'immagine viene scritto all'interno di un file (soluzione immediata e veloce)

- Le funzioni di Input/Output del sistema vengono chiamate da un programma e il segmento di memoria che contiene l'immagine viene salvato
- La memoria sia sul PC (o tablet, smartphone etc) che in un file ha un organizzazione lineare. I Bytes sono indirizzati in modo progressivo. Il codice di un programma mantiene le informazioni che sono necessarie per interpretare questi dati: la dimensione (numero di righe e colonne e indice della terna RGB per immagini a colori)
- L'organizzazione della matrice (le sue dimensioni) verrebbe persa.

# Soluzione 2: Lettura/Scrittura dei dati binari

- Alcuni elementi tecnici che sarebbero ambigui senza una descrizione fornita dai metadati
- E' necessario aggiungere metadati in modo che un'immagine sia interpretabile da altri applicativi e avere informazioni che ne definiscono il contesto tecnico
  - Posso sapere la dimensione complessiva dell'immagine, ma in quante righe e colonne era strutturata?
  - La rappresentazione dei dati binari internamente ai processori ha un'ambiguità dovuta all'*endianness* (vedi slide successiva).
  - La compressione dati è una trasformazione che cerca di ridurre la ridondanza dei dati. Ma che metodo di compressione è stato usato? Con quali parametri di compressione?
  - La disposizione dei pixel salvati segue l'ordine per colonna oppure è stato usato l'*interleaving*?
  - L'identificativo e modello del dispositivo che ha raccolto o manipolato l'immagine
  - La data e ora (timestamp) in cui l'immagine è stata presa
  - Negli smartphone il GPS fornisce la posizione geografica del luogo dove l'immagine è stata presa

# Conclusione

- Abbiamo bisogno di strutturare i dati in modo da
  - Dichiarare la loro organizzazione
  - Dichiarare la loro rappresentazione
  - Corredare i dati di informazione accessorie

# Data Storage and Transmission: Endianness

$$z = \underbrace{00010010}_{12_H \text{ (MSB)}} \cdot 00110100 \cdot 01010110 \cdot \underbrace{01111000}_{78_H \text{ (LSB)}}_B = 12345678_H$$

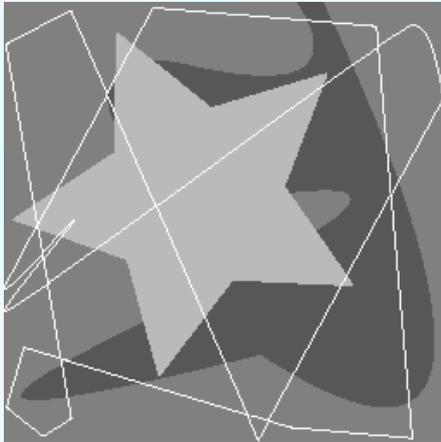
- Un singolo byte può contenere solo numeri tra 0 e 255
- Numeri più grandi o numeri in virgola mobile richiedono più byte
- La parola binaria  $z$  qui sopra rappresentata è composta da 32 bit
- L'espressione più a destra è in esadecimale (pedice H: Hexadecimal): ogni cifra rappresenta un gruppo di 4 bit (riuscite ad identificarli nella parola binaria?), quindi ogni coppia di cifre esadecimale rappresenta un byte
  - Il byte a sinistra con il numero  $12_H (=18_{10})$  è il byte più significativo di questa parola
  - Il byte a destra con il numero  $78_H (=120_{10})$  è quello meno significativo
- Nel corso dell'evoluzione e sviluppo di sistemi di elaborazione digitali i costruttori hanno adottato uno dei due diversi modi per disporre in memoria una simile parola binaria. Queste due convenzioni differiscono per la cosiddetta **endianness**

# Data Storage and Transmission: Endianness

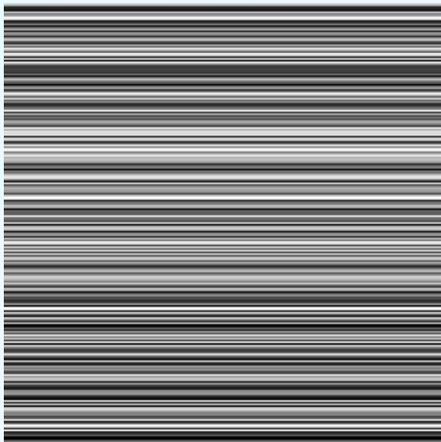
Ordering	Byte Sequence	1	2	3	4
<i>Big Endian</i>	MSB → LSB	12 <sub>H</sub>	34 <sub>H</sub>	56 <sub>H</sub>	78 <sub>H</sub>
<i>Little Endian</i>	LSB → MSB	78 <sub>H</sub>	56 <sub>H</sub>	34 <sub>H</sub>	12 <sub>H</sub>

- Le due endianness alternative sono *big-endian* e *little-endian*
- Dati binari scritti in una convenzione hanno bisogno di essere trasformati per poter essere processati da un processore costruito con l'altra convenzione

# Ridondanza dei Dati



Ridondanza nella  
rappresentazione dei valori



Ridondanza spaziale (nei  
video anche ridondanza  
temporale)

# Compressione Dati

- Un insieme di informazioni codificate in N bit può essere rappresentato in un numero inferiore n di bit con rapporto di compressione che dipendono dalla struttura dei dati e dal metodo di compressione
- Tecnologie basate su diversi metodi matematici, sull'inferenza statistica e tecniche informatiche
- Gli algoritmi di compressione hanno un risultato che viene caratterizzato da
  - Rapporto di Compressione =  $N / n$   
oppure con un parametro equivalente
  - Risparmio di Spazio =  $1 - n / N$

# Compressione Dati

- Esistono 2 classi di algoritmi di compressione dati
  - Lossless: algoritmi di compressione senza perdita di informazione.
    - Riducono la ridondanza permettendo di ricostruire esattamente i dati originari.
    - Indispensabili per compressione di archivi software
  - Lossy: algoritmi di compressione con perdita di informazione.
    - Permettono di ricostruire l'essenza dell'informazione originaria, ma non i dati originari
- ***Usati quando il 'consumatore', cioè il sistema che deve interpretare le informazioni, è un sistema a minore capacità di discriminazione***

# Esempio: Standard Definition Television

- L'importanza degli algoritmi di compressione si può constatare nell'analisi dei dati necessari per memorizzare (ed eventualmente trasmettere) un film. Le caratteristiche tecniche non sono particolari
  - Film in definizione standard (no HD!) della durata di 2 ore
  - Frame (fotogrammi) con la dimensione assai modesta di 720x480 pixel
  - Pixel rappresentati in truecolor (24 bit / pixel)
  - Film ripreso con la frequenza di 30 frame / secondo
  - I dati al secondo necessari sono

$$30 \frac{\text{frames}}{\text{sec}} * (720 * 480) \frac{\text{pixel}}{\text{frame}} * 3 \frac{\text{bytes}}{\text{pixel}} = 31.114.000 \frac{\text{bytes}}{\text{sec}}$$

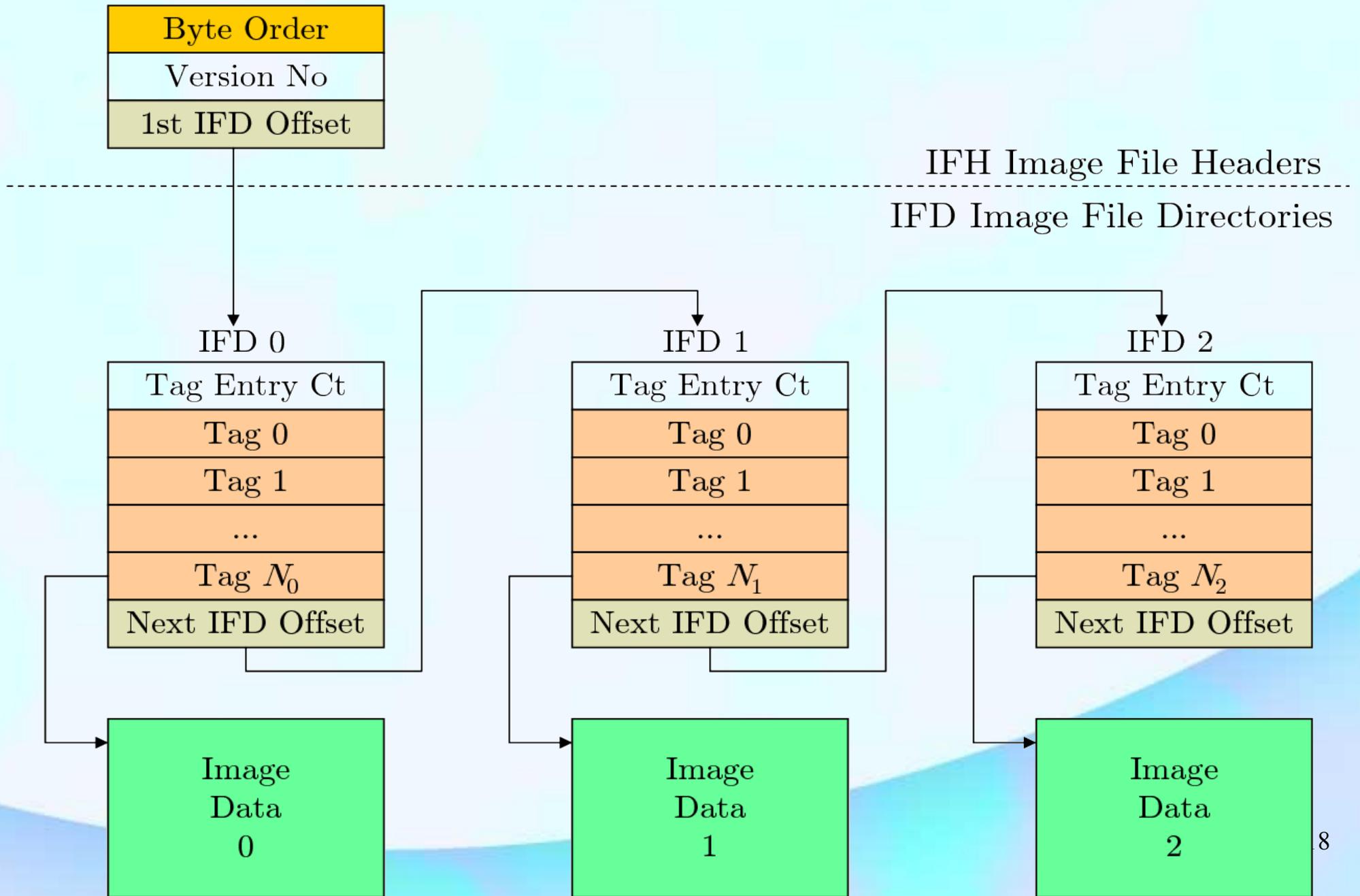
- Un film di 2 ore richiederebbe  $2.20 * 10^{11}$  bytes
- Un DVD double-sided e double-layer può contenere  $\sim 1.8 * 10^{10}$  byte
- Un film di questa lunghezza richiederebbe circa 12 DVD

Alcuni formati di immagine comunemente in uso

# TIFF: Tagged Image File Format

- Supporta formati multipli e diversi metodi di compressione
- Ogni file può contenere rappresentazioni multiple della stessa immagine
- Usato per archiviazione e scambio tra piattaforme e applicazioni differenti
- Ogni file può avere dimensioni considerevoli: inadatto per costruire pagine web

# TIFF: Tagged Image File Format



# TIFF: Tagged Image File Format

- Tag privati
  - Tag con codice numerico  $\geq 32768$  ( $1000_{16}$ ) sono tag definiti da applicazioni specifiche
    - Consorzi
    - Progetti sperimentali
    - Programmatori privati

# GIF: Graphics Interchange Format

- Formato proprietario (Comuserve, 1986)
  - Supporto per immagini “indexed”
    - La pixmap è costituita da indici all'interno di una tavolozza memorizzata nell'header
    - Supporto per bit-depth massima di 8 bit
    - Supporto per bit di trasparenza
  - Compressione LZW (Lempel-Ziv-Welch)
  - Interlaced image loading
  - Possibilità di creare singole semplici animazioni
  - Capacità di compattare lo spazio dei colori per immagini con pochi colori (e.g. icone)

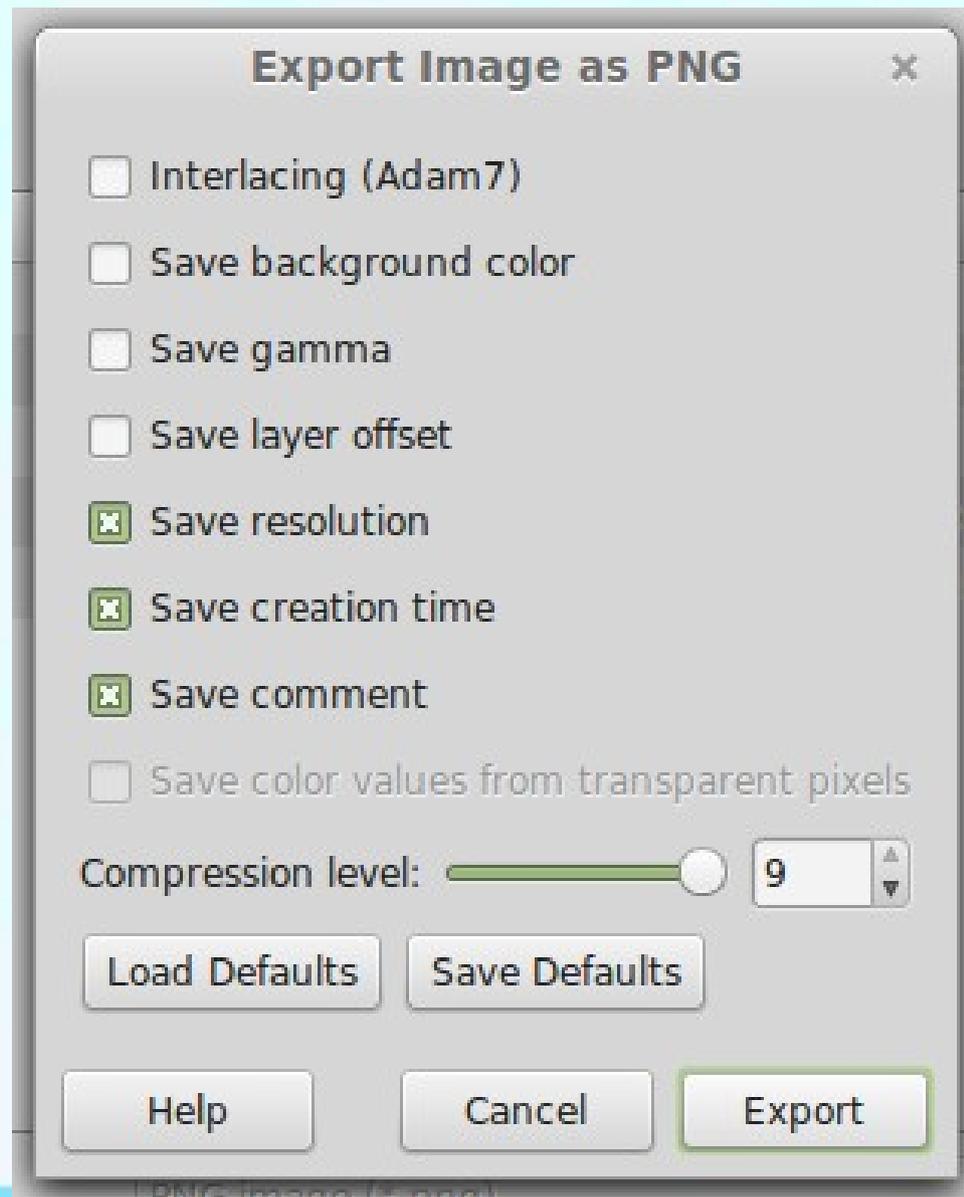
# PNG: Portable Network Graphics

- Pronunciato in inglese 'PING'
- Formato aperto pensato per Internet
  - Truecolor (fino a 48 bit/pixel)
  - Grayscale (fino a 16 bit/pixel)
  - Indexed (fino a 256 colori)
- Alpha Channel 8 bit
- Capacità teorica fino a  $2^{30} \times 2^{30}$  pixel
- Compressione lossless basata su PKZIP

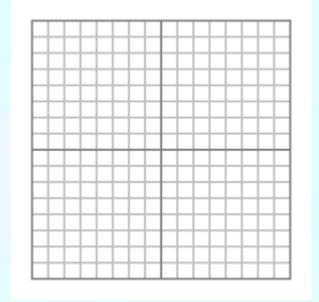
# PNG: Portable Network Graphics

- Ammette estensioni e memorizzazione metadati
  - Data e ora (timestamp)
  - Annotazioni
  - Creazione (data di creazione reale, non quella del file)
  - Copyright and Authorship
  - Colore di background preferito (per applicativi grafici)
  - Tavolozza dei colori (palette) suggerita anche con canale alpha (RGBA)
  - Scala fisica dell'oggetto rappresentato

# PNG: Portable Network Graphics



# Interlacing



- Formazione incrementale dell'immagine
  - L'immagine sembra formarsi e diventare progressivamente più leggibile
  - Permette di avere una percezione preliminare del contenuto dell'immagine
  - Utili su Internet ma in realtà sviluppati per trasmissioni su linee a bassa velocità

# JPEG (Joint Photographic Experts Group)

- Creato nel 1990, diventato un standard ISO
- Design modulare
- Adattabile a diversi tipi di immagine

# JPEG (Joint Photographic Experts Group)

- Caso di manipolazione di immagini RGB
  - Trasformazione da RGB a  $YC_bC_r$  dove  $Y$  rappresenta la luminosità mentre  $C_b$  e  $C_r$  sono la componente cromatica (detta anche *croma*)
  - Compressione con algoritmo 'lossy' con rapporti di compressione differenziati tra croma e luminosità
  - Trasformata cos e quantizzazione nello spazio delle frequenze
  - Ulteriore compressione lossless

# JPEG: RGB $\rightarrow$ $YC_bC_r$

- Sfrutta la trasformazione dallo spazio RGB allo spazio  $YC_bC_r$

$$\begin{pmatrix} Y \\ C_b \\ C_r \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{pmatrix} \cdot \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

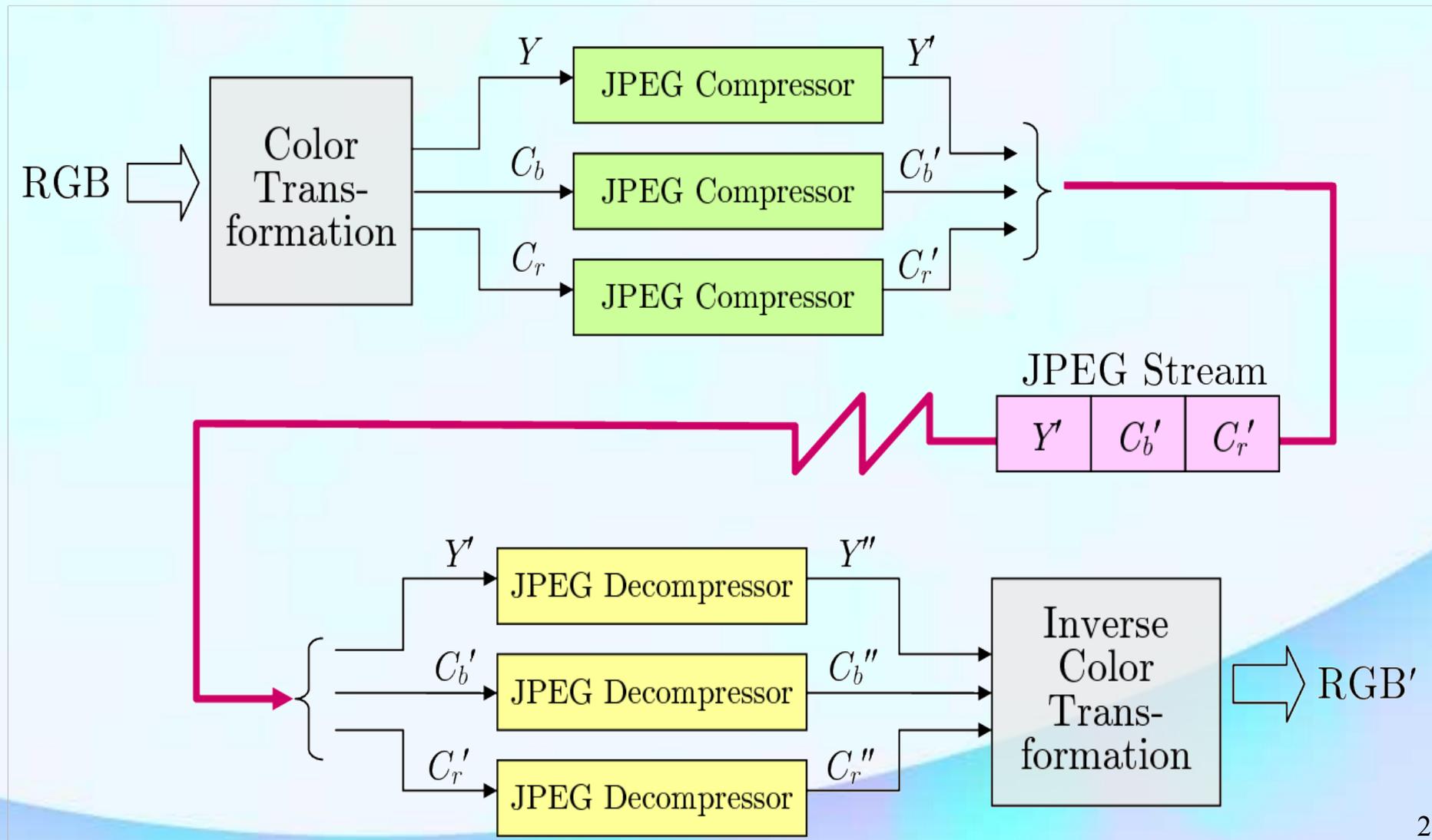
- La componente  $Y$  è la luminosità associata ad una terna RGB secondo lo standard ITU-R BT.601

- Esiste la trasformazione inversa

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.000 & 0.000 & 1.403 \\ 1.000 & -0.344 & -0.714 \\ 1.000 & 1.773 & 0.000 \end{pmatrix} \cdot \begin{pmatrix} Y \\ C_b \\ C_r \end{pmatrix}$$

- L'occhio umano è più sensibile alle variazioni di luminosità che di colore

# JPEG (Joint Photographic Experts Group)



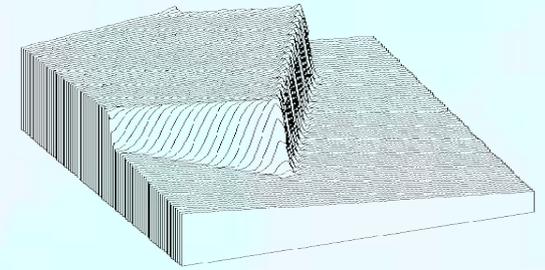
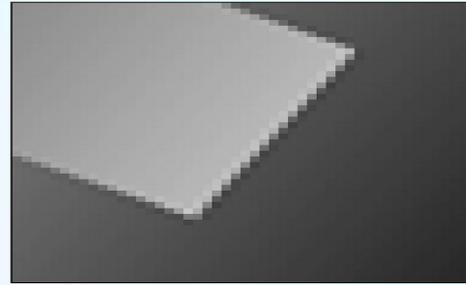
# JPEG (Joint Photographic Experts Group)



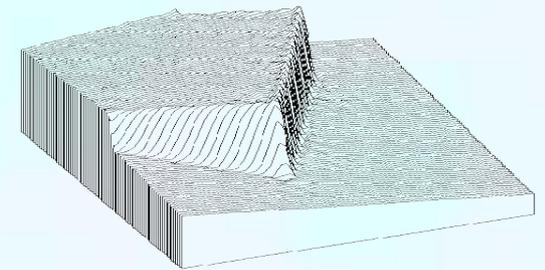
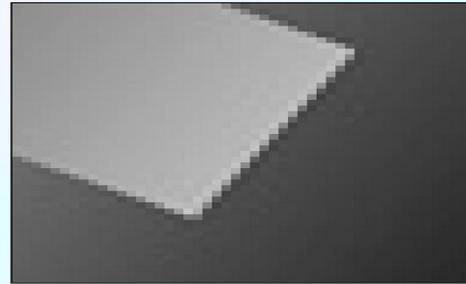
# JPEG (Joint Photographic Experts Group)

- Inadatto per memorizzare su file immagini con molte curve o linee dal contorno netto.

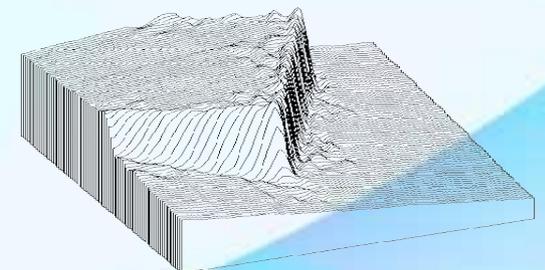
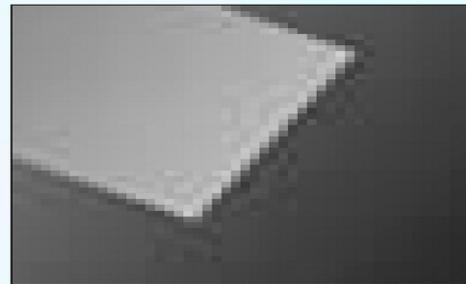
(a) Original  
(75.08 kB)



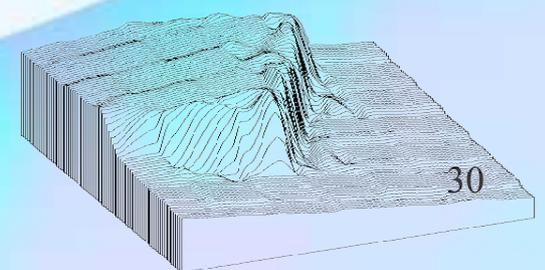
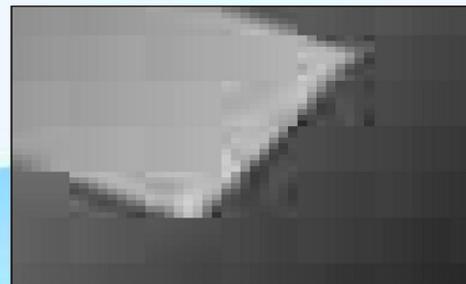
(b)  $Q_{\text{JPG}} = 10$   
(11.40 kB)



(c)  $Q_{\text{JPG}} = 5$   
(7.24 kB)



(d)  $Q_{\text{JPG}} = 1$   
(5.52 kB)



# Windows® Bitmap (BMP)

- Ancora usato in ambiente Windows
  - Può memorizzare immagini grayscale, indexed e RGB
  - Inefficiente perché invariabilmente usa un byte per ogni pixel
  - Supporto basilare per la compressione 'lossless'

# Formati di Immagine

<i>Format</i>	<i>Signature</i>
PNG	0x89504e47    □PNG
JPEG/JFIF	0xffd8ffe0    □□□□
TIFF <sub>little</sub>	0x49492a00    II*□
TIFF <sub>big</sub>	0x4d4d002a    MM□*

<i>Format</i>	<i>Signature</i>
BMP	0x424d    BM
GIF	0x4749463839    GIF89
Photoshop	0x38425053    8BPS
PS/EPS	0x25215053    %!PS

# Best Graphic Format

ACTION	BMP	GIF	JPEG	PNG	TIFF
Post an image on a Web page		X	X		
Export easily to other computer imaging programs	X				X
Compress a large image to create a small file to send in e-mail		X	X		
Use transparent areas in images		X		X	
Create animation files		X			
Decompress images without loss of quality		X		X	
Create highest quality images (for publishing, etc.)					X
Display millions of colors in an image	X		X	X	X
Retain image quality through numerous saves	X	X		X	X