

Appunti di

Interazione e Multimedia

Rosario Terranova

v 1.0.3

Sommario

Introduzione	3
Multimedia	3
Immagini	3
Storia delle immagini	3
Rappresentazione di un'immagine	5
Immagini raster	5
Tipologie di immagini	6
L'occhio	6
Formazione dell'immagine: modello del Pinhole	7
Formazione dell'immagine: modello delle Lenti sottili.....	7
Acquisizione delle immagini digitali.....	9
Natura matriciale delle immagini.....	11
Operazioni affini.....	11
Campionamento e quantizzazione.....	13
Nyquist rate.....	13
Teorema del campionamento di Shannon.....	13
Sottocampionamento	14
Quantizzazione.....	14
Risoluzione	17
Risoluzioni standard.....	17
Ridimensionamento delle immagini	17
Interpolazione	18
Colore	21
Luce	21
Percezione dei colori	21
Standard dei Colori	22
Gli spazi di colore	23
Modello RGB	24
Rappresentazioni luminanza-crominanza.....	25
Colori e memoria	26
Operazioni Puntuali.....	27
L'istogramma	27
Operatori puntuali	27
Aritmetica delle immagini	29
Equalizzazione	30
Bit-plane	30

Convoluzioni e loro proprietà	32
Operatori locali	32
Convoluzione.....	33
Esempi di operatori locali	34
Estrazione dei contorni	35
Serie e Trasformata di Fourier	37
Image Enhancement nel Dominio delle Frequenze.....	37
Trasformata di Fourier	37
Proprietà della DFT 2D	38
Teorema della Convoluzione	39
Filtri low pass nel dominio della frequenza	39
Compressione	40
Algoritmo di Compressione	40
Compressione Lossless.....	40
Compressione Lossy.....	41
Compressione dello standard JPEG.....	41
Matlab.....	46
Cos'è Matlab	46
Rudimenti di Matlab	47

Introduzione

Multimedia



L'aggettivo MULTIMEDIA viene utilizzato per indicare un **sistema** che si avvale di diversi tipi di media. Gli strumenti utilizzati dai media sono i **dispositivi multimediali digitali**.

L'uomo fruisce di tali sistemi multimediali attraverso i suoi sensi (vista, udito e tatto). Tutti i media sono analogici per nascita, occorre quindi trasformarli per avere una versione digitale. La **digitalizzazione** consente a segnali reali di essere convertiti in una sequenza di cifre manipolabili con il computer.

Vantaggi dei media digitali

- Facilmente riproducibili
- Alta qualità
- Si mantengono nel tempo
- Supporti piccoli e facilmente trasportabili
- Interattività (pagine web, tv digitale, videoconferenza, distribuzione di musica su internet)

Contenuti multimediali

Si parla di "contenuti multimediali", in ambito informatico, quando per comunicare un'informazione riguardo a qualcosa ci si avvale di molti media, cioè mezzi di comunicazione di massa, diversi: immagini in movimento (video), immagini statiche (fotografie), musica e testo. Ad esempio, un'encyclopedia multimediale (come Wikipedia), a differenza di una normale encyclopédia cartacea, permette di associare ad ogni voce non solo la sua spiegazione testuale, ma anche fotografie, disegni esplicativi, filmati, suoni, commenti audio ecc.

Grazie alle potenzialità espressivo-comunicative, la multimedialità si è diffusa in ogni settore della cultura e società, dall'educazione al gioco, dalla documentazione allo spettacolo ecc. coinvolgendo ogni forma di comunicazione, anche se viene sempre più riferita o perfino fatta coincidere con i new media e il web, considerati multimediali fin dall'origine e per loro stessa natura.

Oggetti multimediali

Fino a pochi anni fa, multimediali erano comunemente definiti i CD-Rom e/o DVD in cui immagini, testo e suoni, combinati insieme, creavano un unico supporto da "leggere" unicamente su un computer. Oggi la multimedialità è molto di più: Internet e una maggiore capacità di gestire contenuti multimediali hanno trasformato il personal computer in un Mass Medium capace di trasformarsi in TV, radio o telefono, oltre che in libro o in macchina fotografica. Oggi, quindi, per multimedia non si intende più un contenuto legato a un'unica tipologia di supporto, ma un'informazione fruibile, condivisibile e modificabile ovunque e su diversi dispositivi, dal computer al palmare, al telefono, alla LIM (lavagna interattiva multimediale) recentemente entrata a far parte dei nuovi sussidi didattici

Immagini

Un'immagine vale più di mille parole... ma pesa di più in memoria

Un'immagine è una metodica di rappresentazione secondo coordinate spaziali indipendenti di un oggetto o di una scena. Contiene informazioni descrittive riferite all'oggetto, alla scena che rappresenta: un'immagine è quindi una distribuzione (che può essere bi o tri-dimensionale) di un'entità fisica.

La comunicazione visuale è la forma più immediata ed efficace di comunicazione. Il **linguaggio delle immagini** è intrinsecamente indeterminato, evocativo, dotato di segni che assumono valore simbolico in relazione al significato che attribuiamo a ciò che osserviamo o al valore pragmatico degli scopi della comunicazione. Esso è stato dell'uomo sin dalla sue origini biologiche, basti pensare ai primi geroglifici trovate nelle vecchie caverne; il linguaggio scritto appartiene all'uomo solo da pochi millenni.

Siamo circondati da immagini, interpretate dall'occhio in modo che il colore, il movimento e la profondità diventino delle vere e proprie dimensioni aggiuntive rispetto all'informazione iniziale.

Storia delle immagini

Prima fotografia



Ecco il primo documento fotografico della storia (1827); il soggetto è occasionale, un tetto visibile dalla finestra dell'autore, Joseph Nicéphore Niépce (1765- 1833).

La lastra eliografica da lui preparata fu "posta" per 8 ore per permettergli di catturare e imprimere l'immagine.

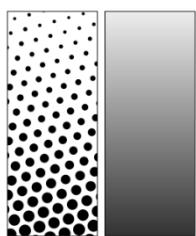
Ma tale fotografia rappresenta comunque un tipo di immagine analogica.

Prima immagine digitale

La prima applicazione di immagine digitale si ha nelle stampe dei quotidiani. Nel 1920 una immagine viene trasmessa via cavo tra New York e Londra al fine di comparire su un quotidiano.

Il protocollo di trasmissione è specifico per l'immagine e il risultato è stampato in **halftoning** da apposite stampanti.

Halftoning



L'halftoning è una tecnica di stampa che simula dei toni differenti di colore attraverso l'uso di punti, che vengono visualizzati diversamente dall'occhio umano a seconda della quantità di inchiostro usato per farne uno, e variando la loro dimensione.

Ad esempio nella figura, a sinistra vediamo dei punti in halftoning, mentre a destra vediamo come l'occhio umano dovrebbe vedere l'insieme di questi punti da una distanza sufficiente a creare questa pseudo-illusione ottica.

Stampa in bianco e nero

La stampa in halftoning è utilizzata per molti anni.

Nel 1922 cambia il tipo di stampa e si possono ottenere fino a 5 livelli di grigio.

Nel 1929 i livelli di grigio diventano 15.

Le sfumature di grigio nelle immagini in bianco e in nero sono dei piccoli puntini di nero, mentre quando i puntini sono più marcati il colore sarà nero scuro.



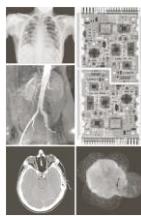
La prima volta che una immagine viene elaborata al computer è nel 1964, quando un computer della NASA riceve ed elabora un'immagine della luna e ne corregge alcune distorsioni ottiche.

Utilizzi professionali delle immagini

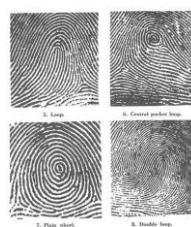
Dal satellite



In medicina



Nelle forze dell'ordine



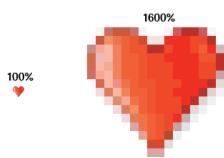
Per i beni culturali



Rappresentazione di un'immagine

La prima differenza di cui bisogna parlare in fatto di immagini è sicuramente quella tra immagini raster ed immagini vettoriali:

Immagini raster



Le immagini raster sono quelle a cui più spesso siamo abituati e sono formate da un reticolo di pixel accostati l'uno all'altro ed a ciascuno di essi è assegnato un colore. La maggior parte delle immagini che si trovano su internet sono appunto immagini raster (GIF, JPG, BMP etc..). Il principale difetto di questo tipo di immagini è che esse perdono qualità e dettaglio se vengono ingrandite presentando il tipico effetto "pixellato". Le immagini realizzate con Photoshop sono quasi sempre immagini raster.

Immagini vettoriali



Le immagini vettoriali invece sono costituite da vettori matematici, più semplicemente da una serie di tracciati e punti che formano figure geometriche. Il vantaggio di questo tipo di immagini è che possono essere ingrandite quanto si vuole senza che presentino mai una perdita di qualità, d'altro canto purtroppo trattandosi solo di linee e punti non è possibile gestire file molto complessi, come ad esempio delle fotografie.

Questo tipo di grafica viene per esempio utilizzato dalle aziende per creare loghi che dovranno essere riprodotti sia in dimensioni ridotte (come su biglietti da visita o carta da lettere) che su grandi dimensioni (come ad esempio su un furgone o uno striscione).

Rappresentazione matematica delle immagini

- **Raster:** nella coordinata x, y c'è un pixel rosso; nella coordinata $x+1, y+1$ c'è un pixel verde, ecc.
- **Vettoriale:** c'è un segmento che unisce il pixel x al pixel y , un altro segmento unisce altri pixel, ecc.

Immagini raster

Un'immagine raster è una funzione bidimensionale $f(x, y)$ dove le due variabili rappresentano un punto.

Il sistema di riferimento non è cartesiano poiché l'origine del sistema delle coordinate $(0,0)$ è il puntino in alto a sinistra.

$f(x, y)$ è proporzionale alla luce incidente nell'oggetto e anche quella riflessa, quindi

$$f(x, y) = i(x, y) r(x, y)$$

Dove i è la luce incidente ed r quella riflessa.

Range dei valori i ed r

$$\begin{aligned} 0 < i(x, y) &< \infty \\ 0 < r(x, y) &< 1 \end{aligned}$$

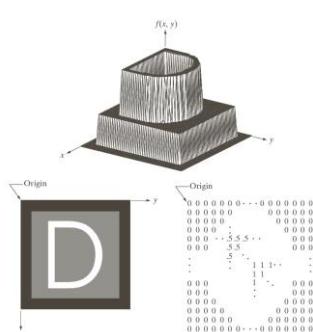
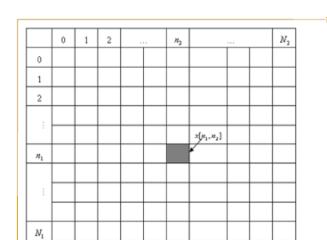
Es. i : la luce del sole può produrre più di 90000 lm/m^2 di illuminazione sulla superficie terrestre. La luna piena raggiunge 0.1 lm/m^2 di illuminazione

r : 0.01 per il velluto nero, 0.80 per un muro verniciato di bianco, 0.90 per l'argento e 0.93 per la neve.

Convenzioni

È importante ricordare che convenzionalmente il primo elemento della matrice è sempre l'elemento in alto a sinistra. L'asse a destra è orientato dall'alto verso il basso; l'asse in alto è orientato da sinistra verso destra. Questo è il quarto quadrante del piano cartesiano, ma le coordinate sono tutte positive.

In teoria il valore di $f(x, y)$ è un numero reale, ma per produrre un'immagine digitale abbiamo bisogno di valori discreti. Questo passaggio dal continuo al discreto è fatto mediante le operazioni di **campionamento** e di **quantizzazione**.



Un **pixel** (picture element) è il valore quantizzato misurato da ciascun sensore di cattura delle immagini.

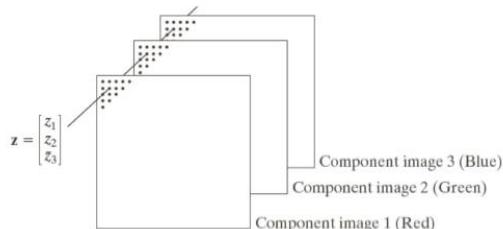
Nell'immagine di accanto possiamo vedere la differenza tra l'immagine originale, la stessa immagine con i suoi valori dei pixel vista in 2 dimensioni, e la stessa immagine con i valori dei suoi pixel vista in 3 dimensioni.

Tipologie di immagini

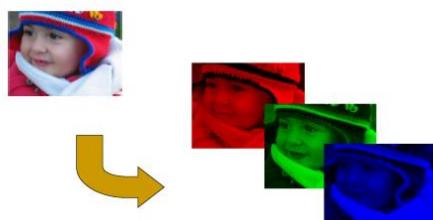
<u>Immagini binarie:</u>	Bianco/Nero, 1 bit per pixel. Nella posizione (i,j) ci sarà o il valore 0 o il valore 1.	
<u>Toni grigio:</u>	8 bit per pixel (1 byte). Nella posizione (i,j) ci sarà un valore compreso tra [0,255] poiché $2^8 = 256$.	
<u>Colori:</u>	Usano 3 byte (per la terna RGB). 8 bit per canale, poiché i canali sono 3 avrà 24 bit. Nella posizione (i,j) ci sarà una terna del tipo (x,y,z) con x,y,z che assumono valori compresi tra [0,255]. In totale 2^{24} colori (16777216 colori).	

Immagini RGB (rosso, verde, blu)

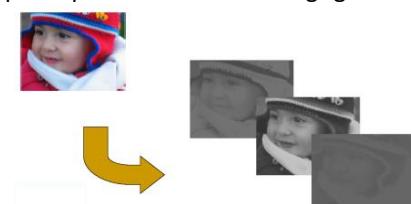
Abbiamo 3 livelli per un'immagine RGB: nel primo le coordinate del rosso, nel secondo del verde, nel terzo del blu.



Siamo tentati di dire che ogni livello separato dagli altri ha il colore predefinito dello schema RGB come in figura



Ma in realtà ogni livello ha solamente 8 bit, quindi presenta una scala di grigi.



Lo studio dell'image processing inizia necessariamente con lo studio dell'occhio umano, il modo in cui percepisce un'immagine e il modo come la elabora. Ci interessa capire quali sono i limiti della visione umana al fine di usarli nell'image processing.

L'occhio

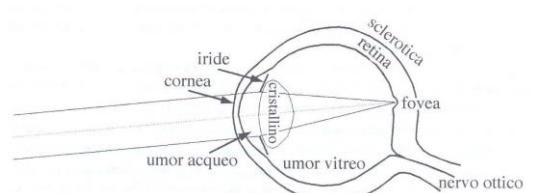
La parte che a noi interessa dell'occhio è la **retina**, una membrana che ricopre la parte posteriore dell'occhio. Essa è formata da coni e bastoncelli che sono i **fotorecettori**.

Coni

Sono circa 6/7 milioni, concentrati nella **fovea**, una zona centrale della retina.

Sono fortemente sensibili al colore ed ogni cono è collegato ad un nervo ottico.

Essi sono responsabili del colore, la vista **fototica** o policroma. Senza coni vedremo solo in scala di grigio, senza colori.



Bastoncelli

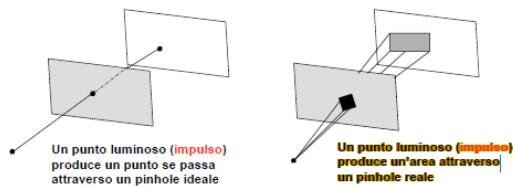
Sono circa 75/150 milioni, concentrati su tutta la retina. Sono poco sensibili al colore e sono collegati a gruppi ad un nervo ottico. Sono responsabili della visione **monocromatica**, o scotopica.

Fovea

La fovea ha una grandezza di 1,5 mm x 1,5 mm ed ha circa 337 mila coni. Un CCD (rivelatore di luce elettronico, un sensore della macchina fotografica) può contenere lo stesso numero di recettori in non meno di 5 mm x 5 mm.

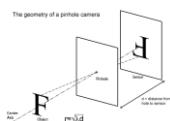
Formazione dell'immagine: modello del Pinhole

Per comprendere come si forma l'immagine nell'occhio occorre astrarre il problema e considerare il modello del **PINHOLE**. Si tratta di un modello teorico in cui si approssima l'occhio con una scatola; all'interno della scatola, su una parete, viene posizionata una pellicola sensibile alla luce. Nella parete opposta si pratica un foro con uno spillo (pinhole).



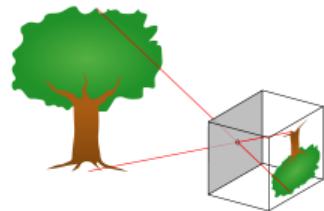
Un foro puntiforme senza estensione non fa passare un numero sufficiente di fotoni per attivare i sensori e produrre un area; un pinhole reale invece presenterà un foro con una precisa forma geometrica (figura a sinistra).

impulso) su una area finita. Questa "macchia" generata dallo spargersi di un punto luminoso si chiama **POINT SPREAD FUNCTION** del sistema di acquisizione di immagini.



Il raggio del foro è proporzionale alla radice quadrata della distanza per la lunghezza d'onda della luce emessa.

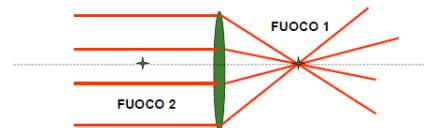
$$r = \sqrt{\lambda d}$$



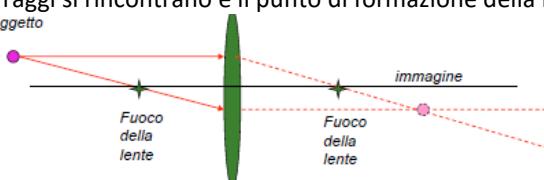
Formazione dell'immagine: modello delle Lenti sottili

Nella pratica i forellini del pinhole sono inadeguati: raccolgono troppe poche radiazioni per consentire ai sensori misurazioni precise. La vera approssimazione dell'occhio viene fatta con le lenti sottili, aventi proprietà simili al pinhole. Una lente sottile è definita da una proprietà geometrica importante che si può enunciare come due parti "speculari" l'una all'altra:

- raggi paralleli all'asse della lente sottile vengono concentrati in un unico punto detto **FUOCO**, posto a distanza F dalla lente;
- raggi che si dipartono dal FUOCO vengono ri-trasmessi tutti paralleli nella direzione dell'asse della lente.
- una lente sottile ha due fuochi equidistanti da essa.

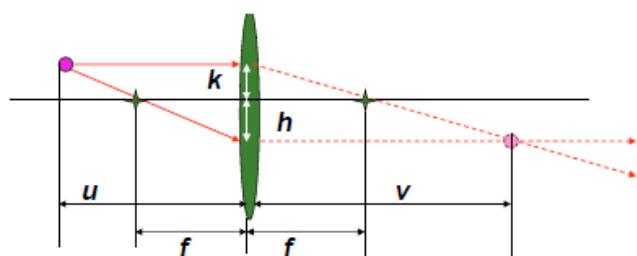


Un oggetto puntiforme, emette raggi luminosi in ogni direzione, solo uno è parallelo all'asse ottico e la lente lo farà passare per il fuoco interno alla lente, mentre solo un altro raggio passa per il fuoco esterno alla lente, ed essa lo farà passare in un raggio parallelo alla lente. Il punto in cui i due raggi si incontrano è il punto di formazione della immagine dell'oggetto puntiforme.



Se si pone il piano dei sensori più avanti o più indietro del piano che contiene l'immagine si ottiene una immagine SFOCATA dell'oggetto originale.

Calcolo dell'equazione della lente sottile



Esiste una relazione che lega tra loro u (distanza dall'oggetto), v (distanza dalla lente allo schermo) ed f (fuoco). Ci aiuteremo con due quantità h e k come in figura.

Il triangolo di base u e altezza $h+k$ e quello di base f e altezza h (entrambi a sinistra della lente) sono simili, da cui:

$$u: (h + k) = f: h \quad \text{dalla quale ricaviamo} \quad (h + k) = \frac{uh}{f}$$

Il triangolo di base v e altezza $h+k$ e quello di base f e altezza k (entrambi a destra della lente) sono simili, da cui:

$$v: (h + k) = f: k \quad \text{dalla quale ricaviamo} \quad (h + k) = \frac{vk}{f}$$

Eguagliando le due relazioni ed eliminando f si ottiene che

$$\frac{uh}{f} = \frac{vk}{f} \quad \rightarrow \quad uh = vk \quad \rightarrow \quad h = \frac{vk}{u} \quad \rightarrow \quad \left(\frac{1}{v}\right)h = \frac{vk}{u} \left(\frac{1}{v}\right) \quad \rightarrow \quad \frac{h}{v} = \frac{k}{u}$$

Ora da $(h + k) = \frac{uh}{f}$ possiamo dividere ambo i membri per u ed otteniamo $\frac{h}{u} + \frac{k}{u} = \frac{h}{f}$ ed ancora sostituendo con l'eguaglianza trovata prima $\frac{h}{v} = \frac{k}{u}$ abbiamo $\frac{h}{u} + \frac{h}{v} = \frac{h}{f}$ da cui eliminando il fattore comune h si giunge a $\frac{1}{u} + \frac{1}{v} = \frac{1}{f}$.

Equazione della lente sottile

$$\frac{1}{u} + \frac{1}{v} = \frac{1}{f}$$

Se f si misura un metri, la quantità $\frac{1}{metro}$ dell'uguaglianza dell'equazione si definisce pari ad una **dioittria**.

In una lente "fissa" la quantità f è costante. Se la distanza dell'oggetto dalla lente, cioè u , cresce, per la relazione di cui sopra, v non può che diminuire: ecco perché la **messa a fuoco** richiede che il piano dei sensori possa essere avvicinato o allontanato dalla lente.

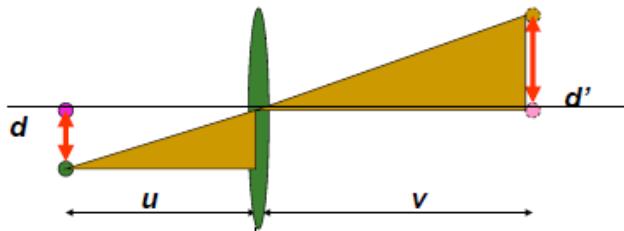
Se il piano dei sensori non può essere mosso (caso della retina umana) l'unica cosa da fare è **aggiustare la lunghezza focale** (ciò fanno i muscoli che mantengono in tensione il cristallino). La variabilità della lunghezza focale della lente si misura anche essa in dioittrie.

Se due oggetti sono a distanza u_1 e u_2 e entrambe queste quantità sono molto maggiori di f essi formano le loro immagini approssimativamente su un unico piano (i due valori corrispondenti v_1 e v_2 sono vicinissimi). Se u_1 e u_2 sono però differenti e comparabili (meno di 30 volte la distanza della lente) allora essi non possono essere focalizzati contemporaneamente: si manifesta il fenomeno della "**profondità di campo**" che risulta più accentuato se f è grande.

Magnificazione (o ingrandimento)

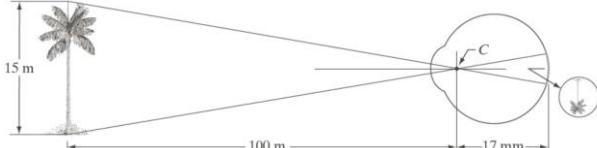
La magnificazione è il processo che aumenta le dimensioni di un oggetto a livello ottico e non a livello fisico. Essa è anche un numero che descrive con quale fattore sia stato ingrandito un oggetto. Quando questo numero è minore di uno si riferisce ad una riduzione del formato, chiamata rimpicciolimento.

Per scoprire come vengono trasformate le distanze dalla lente sottile dobbiamo studiare la relazione tra d (misura dell'oggetto) e d' (misura della sua immagine).



Poiché si può dimostrare che i due triangoli sono simili, si ha facilmente che $\frac{d'}{d} = \frac{v}{u} = m$, con m che è il fattore di magnificazione. Partiamo dalla equazione della lente sottile $\frac{1}{u} + \frac{1}{v} = \frac{1}{f}$ moltiplico per v ottenendo $\frac{v}{u} + 1 = \frac{v}{f}$ e poi sostituisco con m ed invertendo $\frac{f}{v} = \frac{1}{m+1}$, moltiplicando per u si ha $\frac{f}{m} = \frac{u}{m+1}$ da cui si giunge a $f = \frac{um}{m+1}$.

Es.



Nella figura a sinistra $v = 17mm$, $h = 15m$ e $u = 100m$. Per il principio della magnificazione si ha che l'altezza h dell'immagine rimpicciolita sulla retina è

$$\frac{15}{100} = \frac{h}{17} \quad \text{cioè } h = 2,55mm$$

La relazione $f = \frac{um}{m+1}$ è utile se si vuole fissare il fuoco in modo da garantire una magnificazione fissata.

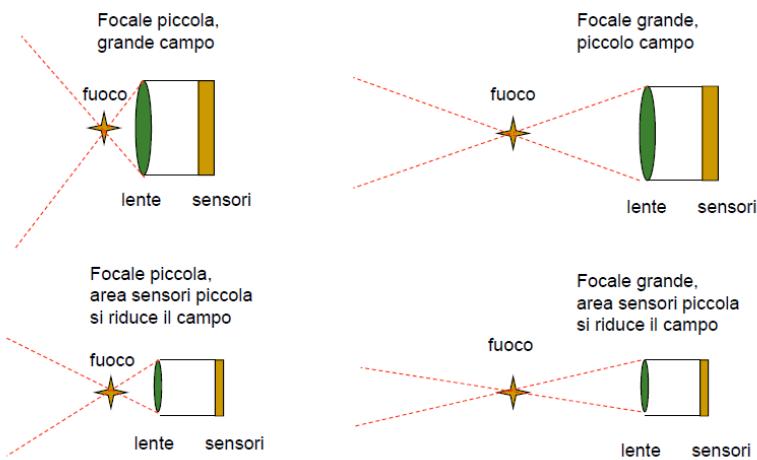
Es.



Il ragno distante $u = 1cm$ è fotografato da una macchina digitale con chip CCD distante $v = 0,5cm$. A quanto si deve fissare il fuoco per potere avere lo stesso effetto su una farfalla che si trova a $u' = 500cm$ dalla lente?

Il fattore di magnificazione è $m = \frac{0,5}{1} = 0,5$ quindi $f = \frac{500 \cdot 0,5}{1,5} = 166,66\text{ cm}$

Ampiezza di campo e focale



Intensità percepita, range e illusioni ottiche

Poiché le immagini digitali sono rappresentate da un numero finito di intensità, è importante conoscere come l'occhio umano riesca a discriminare tra i diversi livelli di intensità. Si è sperimentalmente dimostrato che l'intensità percepita è funzione logaritmica dell'intensità incidente nell'occhio.

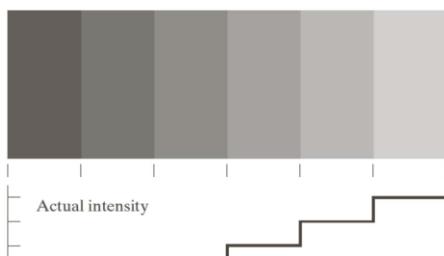
Il sistema visivo umano non opera contemporaneamente su tutto il *range delle intensità percepite*, ma solo su una porzione di esso. Inoltre riesce a distinguere in maniera differente se si trova in una zona chiara o in una zona scura.

Esistono dei fenomeni di *illusioni ottiche* che sono tipiche del nostro sistema visivo umano e che non sono ancora state spiegate. Ciò sta ad indicare che luminosità percepita non è semplicemente in funzione dell'intensità emessa.

Esempi:

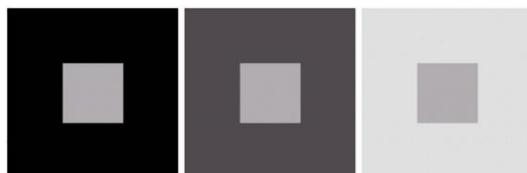
Bande di Mach

Anche se le bande hanno una intensità costante, esse vengono percepite in maniera non uniforme all'approssimarsi dei bordi.

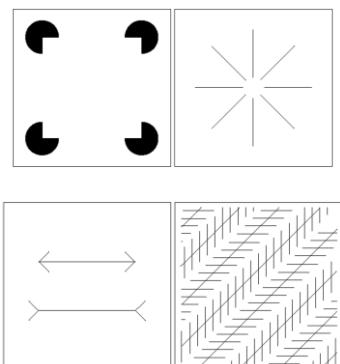


Contrasto simultaneo

Anche se le intensità nei quadratini centrali sono uguali, esse vengono percepiti in maniera differente in base allo sfondo: se lo sfondo è scuro, il centro appare più chiaro e viceversa.



Altre illusioni ottiche



Acquisizione delle immagini digitali

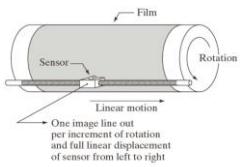
Dicembre 1975, Steven Sasson, un ingegnere elettronico dell'Eastman Kodak Co., nel Rochester, NY, diventa la prima persona a catturare un'immagine digitale.

Come viene catturata un'immagine

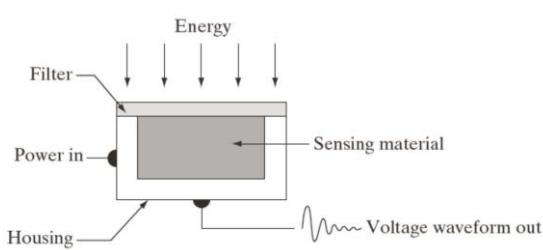
Fisicamente, quando la luce colpisce un oggetto, una parte viene assorbita ed una parte viene riflessa. Quella che viene riflessa, dà origine al colore percepito. Per creare una immagine digitale, è essenziale che tale luce riflessa sia catturata da un **sensore** ed elaborata.

Il sensore

L'energia che colpisce il sensore è trasformata in impulso elettrico dal sensore stesso che è fatto di un materiale particolarmente sensibile alla luce. Tale impulso elettrico è successivamente digitalizzato.



Gli scanner digitali usano un singolo sensore che viene spostato lungo la sorgente da digitalizzare (foglio o foto).

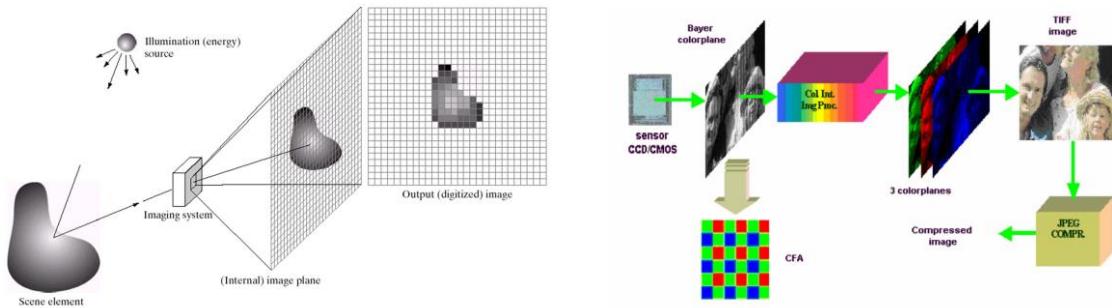


Nelle macchine fotografiche digitali invece i sensori sono disposti su una matrice. Non è necessario spostare il sensore, come nello scanner, per effettuare una scansione. I più diffusi sensori di questo tipo sono i **CCD**.

CCD (Charge-Coupled Device)

Consiste in un circuito integrato formato da una griglia di elementi semiconduttori in grado di accumulare una carica elettrica proporzionale all'intensità della radiazione elettromagnetica che li colpisce.

Inviando al dispositivo (device) una sequenza temporizzata d'impulsi, si ottiene in uscita un segnale elettrico grazie al quale è possibile ricostruire la **matrice dei pixel** che compongono l'immagine proiettata sulla superficie del CCD stesso.



Gli CCD sono in poche parole dei dispositivi elettronici che se colpiti da fotoni assumono una carica positiva. Essi non possono caricarsi oltre un certo limite di intensità luminosa acquisita (fenomeno della sovra-saturazione), e il numero di celle per area di esposizione è un parametro di qualità della fotocamera misurato in **Megapixel**.

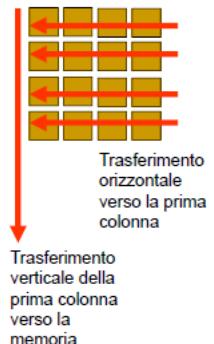
CCD: Memorizzazione delle informazioni

Dopo che le cariche sono state acquisite da una matrice di celle esse debbono essere trasferite in una memoria digitale. La scansione avviene in *C* fasi, una fase per ciascuna colonna della matrice.

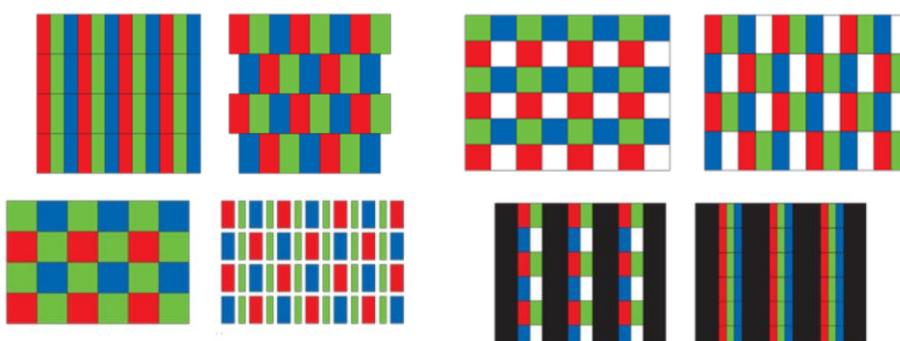
Ad ogni fase viene trasferita in memoria la prima colonna della matrice, nello stesso tempo tutti gli elementi (dalla seconda colonna in poi) vengono trasferiti dalla propria colonna a quella precedente.

Poiché ogni cella memorizzerà solo un colore per volta e non una terna, occorre scegliere qual è il modello di memorizzazione ottimale.

I due colori mancanti per completare la terna, saranno ottenuti per **interpolazione** (metodo matematico per individuare nuovi punti del piano cartesiano a partire da un insieme finito di punti dati) dai pixel vicini. Il grado di accuratezza del risultato dipende da quanto è sofisticato il metodo di interpolazione.



Possibili modelli di memorizzazione



Esempio di applicazione di modelli diversi



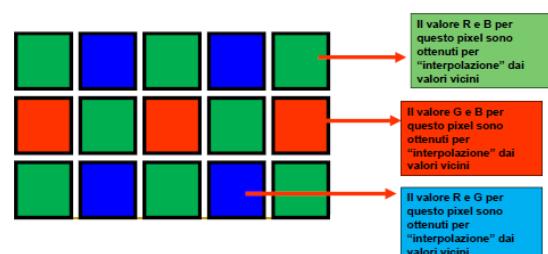
+ sharpening

Bayer Pattern



Lo schema migliore è il **Bayer Pattern**, utilizzato dal 1980 in tutti i dispositivi elettronici. Esso presenta un rapporto 1:2:1 per R:G:B dove i pixel verdi sono disposti sulle diagonali. Esso privilegia le misure nel canale verde perché è quello più importante per la percezione umana. Un'immagine in Bayer Pattern è conservata nel formato *raw*.

Se per ogni pixel si memorizza solo una componente di colore, tutte le altre dovranno essere ottenute per interpolazione dai pixel vicini. Esistono diversi tipi di interpolazione (replicazione, bilineare, bicubico, ecc.) e la loro differenza sta nell'algoritmo usato per calcolare di che colore deve essere il pixel adiacente.



Natura matriciale delle immagini

Dato che, dunque, l'immagine acquisita da un CCD è una matrice, e ogni singolo elemento della matrice è un PIXEL, su un'immagine possono essere fatte tutte le operazioni che si possono fare sulle matici.

Prodotto puntuale

Per le matrici vale la regola del **prodotto riga per colonna**, mentre nell'image processing si usa fare il **prodotto puntuale** tra due matrici, cioè il prodotto punto a punto degli elementi corrispondenti.

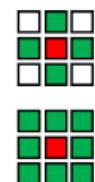
$$\text{Es. } A \times B = \begin{bmatrix} 1 & 0 & 2 \\ -1 & 3 & 1 \end{bmatrix} \times \begin{bmatrix} 3 & 1 \\ 2 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} (1 \times 3 + 0 \times 2 + 1 \times 1) & (1 \times 1 + 0 \times 1 + 2 \times 0) \\ (-1 \times 3 + 3 \times 2 + 1 \times 1) & (1 \times 1 + 3 \times 1 + 1 \times 0) \end{bmatrix} = \begin{bmatrix} 5 & 1 \\ 4 & 2 \end{bmatrix}$$

$$A \cdot B = \begin{bmatrix} 1 & 2 \\ 3 & -1 \end{bmatrix} \cdot \begin{bmatrix} -3 & 0 \\ 1 & 4 \end{bmatrix} = \begin{bmatrix} -3 & 0 \\ 3 & -4 \end{bmatrix}$$

Neighborhood (vicini) di un punto

I vicini 4 connessi di un dato pixel sono quelli alla sua destra e sinistra e quelli sopra e sotto.

I vicini 8 connessi sono quelli 4 connessi a cui si aggiungono i 4 pixel in diagonale.



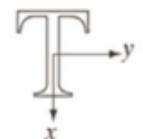
Operazioni affini

Dato (u, v) il pixel di input, (x, y) quello di output e T la matrice affine:

Identità

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned} x &= v \\ y &= w \end{aligned}$$



Riscalaggio

$$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned} x &= c_x v \\ y &= c_y w \end{aligned}$$



Rotazione

$$\begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

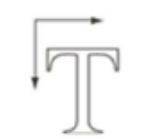
$$\begin{aligned} x &= v \cos \theta - w \sin \theta \\ y &= v \sin \theta + w \cos \theta \end{aligned}$$



Traslazione

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$$

$$\begin{aligned} x &= v + t_x \\ y &= w + t_y \end{aligned}$$



Riflessione verticale

$$\begin{bmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned} x &= v + s_v w \\ y &= w \end{aligned}$$



Riflessione orizzontale

$$\begin{bmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned} x &= v \\ y &= s_h v + w \end{aligned}$$



E.s.

	0	1	2
0	5	3	7
1	6	2	1
2	5	4	3

u e v input
 x e y output

traslazione di 2

$$\begin{aligned} x &= v + tx & v &= 2 \\ y &= w + ty & w &= 2 \end{aligned}$$

$tx = \text{indice } x$
 $ty = \text{indice } y$

$$\begin{aligned} 00: x &= 2 + 0 = 2 \\ y &= 2 + 0 = 2 \end{aligned}$$

$$\begin{aligned} 01: x &= 2 + 0 = 2 \\ y &= 2 + 1 = 3 \end{aligned}$$

	0	1	2	3
0				
1				
2			5	3

Forward e inverse mapping

Le formule per eseguire le operazioni affini su un'immagine sono di due tipi

- *Forward Mapping:* $[x \ y \ 1] = [u \ w \ 1] * T$
- *Inverse Mapping:* $[x \ y \ 1] = [u \ w \ 1] * \text{inversa}(T)$

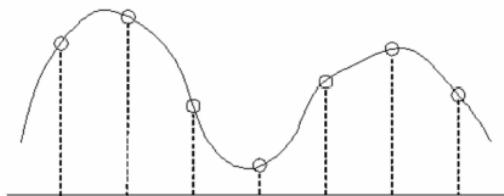
Il forward mapping e l'inverse mapping sono dei processi che permettono di applicare le formule delle operazioni affini a tutti i pixel di una matrice d'immagine. In particolare, con il **forward mapping** si fa scorrere l'immagine di input e per ogni suo pixel (v, w) viene calcolata la sua posizione nell'immagine di output (x, y) ; con l'**inverse mapping** invece si visitano le posizioni spaziali dei pixel dell'immagine di output (x, y) e si calcolano le corrispondenti coordinate nell'immagine di input (v, w) (si ha una formula inversa).

Nel corso delle trasformazioni, potrebbero esserci dei valori di pixel che non sono mai individuati dalle formule; per essi si applica un processo di interpolazione.

Campionamento e quantizzazione

Un'immagine digitale è un insieme di numeri interi, ottenuti dalla scansione di un'immagine analogica (sorgente) tramite processo di **digitalizzazione**, fatto utilizzando un'apparecchiatura speciale detta scanner o tramite l'utilizzo di fotocamere digitali che producono direttamente l'immagine digitale dalla scena ripresa.

Un'immagine analogica è un segnale continuo e limitato; nella teoria dei segnali il **campionamento** è una tecnica che consiste nel convertire un segnale continuo nel tempo in un **segnale discreto** (funzione, o un segnale, con valori forniti in corrispondenza ad una serie di tempi scelti nel dominio dei numeri interi), valutandone l'ampiezza a intervalli di tempo regolari. In questo modo, a seguito di una successiva operazione di **quantizzazione** e conversione, è possibile ottenere una **stringa digitale** (discreta nel tempo e nell'ampiezza) che approssimi quella continua originaria.



In parole poche il campionamento consiste nell'andare a "sentire" (misurare, registrare) il valore del segnale analogico in diversi istanti di tempo.

Dato un segnale continuo occorre scegliere un numero finito di "campioni" rappresentativi del segnale. Il valore in ogni singolo punto del segnale è un numero reale, occorre scegliere dei valori discreti per rappresentare correttamente il segnale.

Errore nel tasso di campionamento

- Un campionamento troppo basso fa perdere dettagli ed informazioni; sebbene grave una tale perdita è spesso una necessità: non possiamo conservare milioni di campioni e ci accontentiamo di perdere informazioni pur di tenere il database delle misure ottenute in dimensioni maneggevoli.
- Un campionamento troppo basso può far apparire nella immagine dettagli NON PRESENTI nell'originale; il segnale viene "alterato" e cambiato in qualcosa di "altro". Si parla di **"aliasing"**. L'aliasing è un fenomeno sottile ma poiché esso è imprevedibile richiede attenzione.

Per scegliere il giusto valore di campionamento ed evitare questi errori si ricorre ad un teorema fondamentale: il **teorema di Shannon**. Tale teorema si basa sulla misura della **frequenza di Nyquist**.

Nyquist rate

Si definisce Nyquist rate la più alta frequenza in un segnale continuo e limitato.

Si osservi un fenomeno che si svolge in un intervallo a...b

- Se il fenomeno è costante durante tutto l'intervallo, la frequenza di Nyquist è 1: il fenomeno si svolge in un unico ciclo.
- Altrimenti si divide l'intervallo in 2 parti e si controlla se per ciascun intervallino il fenomeno si mantiene costante (esso può però variare da intervallino ad intervallino).
- Si procede in tal modo dividendo l'intervallo in 3, 4, ... parti fino a trovare una suddivisione tale che entro ciascun intervallino il fenomeno sia in pratica costante.

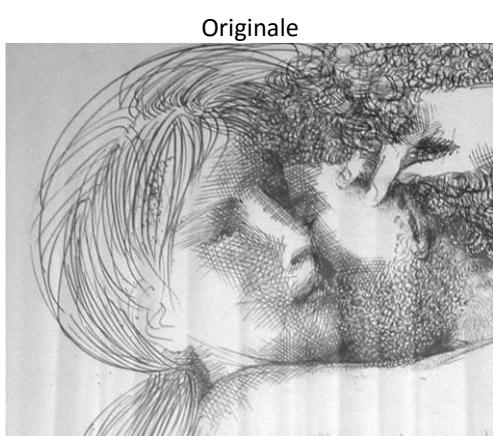
Sia tale suddivisione in N parti. N si dice frequenza di Nyquist del fenomeno sull'intervallo osservato.

Teorema del campionamento di Shannon

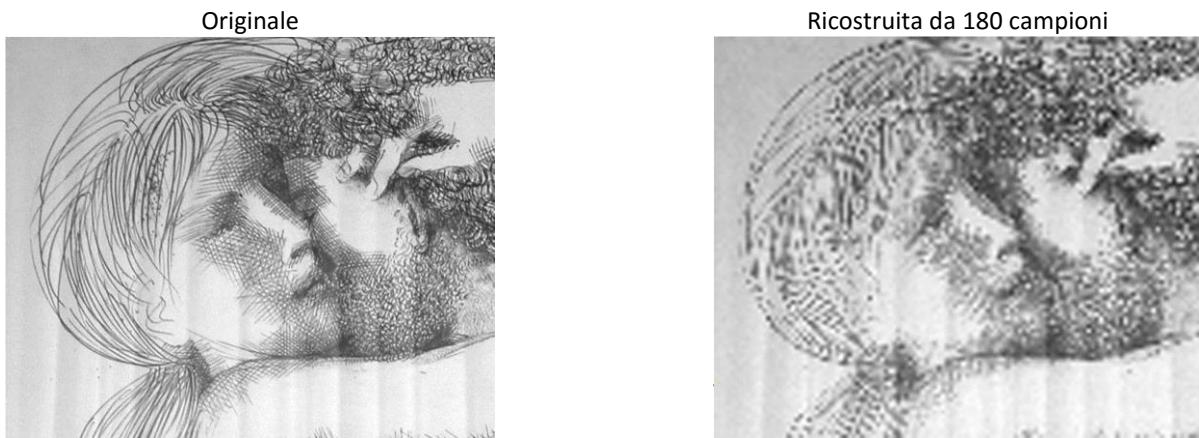
Se si raccolgono campioni con frequenza almeno doppia della frequenza di Nyquist (**2N** nel nostro caso) il segnale può essere ricostruito **FEDELMENTE** in ogni suo punto!

Es. Vediamo come agisce il campionamento attraverso un'immagine in scala di grigi usando i tratti fini:

- dimensione quadro 720 pixel, dettaglio massimo 4 pixel, possiamo dividere l'intervallo in $720/4=180$ tratti.
- Il doppio della frequenza di Nyquist è 360. Prenderemo allora solo 360 campioni e ricostruiremo con l'interpolazione binomiale l'immagine.



Campionamento NON al doppio della frequenza di Nyquist



Es. Devo fotografare dal satellite un terreno di 1000x5000 pixel con un dettaglio massimo di 3 pixel, di quanti campioni ho bisogno?

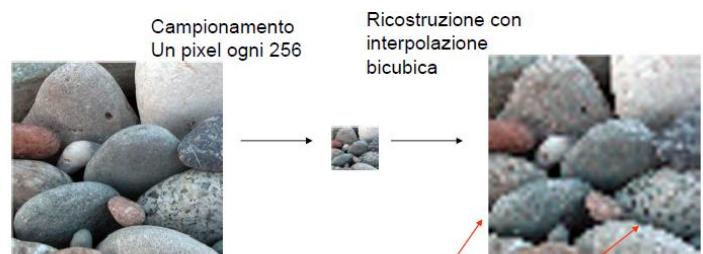
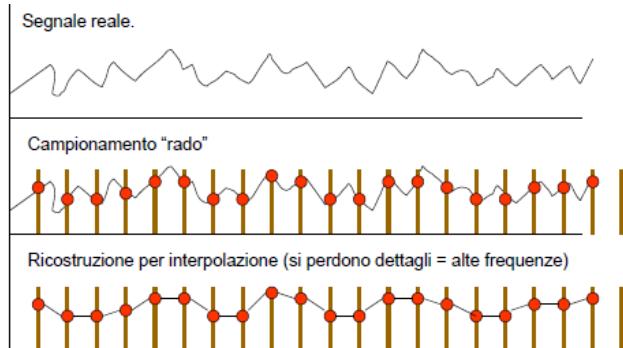
$$\left(\frac{1000}{3} * 2\right) * \left(\frac{5000}{3} * 2\right) = 3999 \text{ campioni}$$

Sottocampionamento

Se si campiona ad una frequenza inferiore a quella di Nyquist si perdono dei dettagli significativi e spesso si introducono nuovi dettagli che non sono presenti nella realtà. Questo fenomeno è detto *frequency aliasing* o semplicemente *aliasing*.

Aliasing

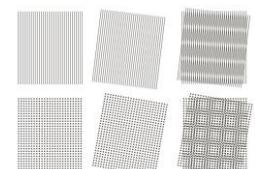
Con l'aliasing le alte frequenze sono "mascherate" da basse frequenze e trattate come tali nella fase di campionamento.
Aliasing proviene da Alias cioè falsa identità!



Nella realtà l'aliasing è sempre presente anche se in condizioni minime. Esso viene introdotto quando si impone che il segnale sia limitato per essere campionato. L'aliasing può essere ridotto applicando una funzione di smussamento sul segnale originario prima del campionamento (**antialiasing**).

Effetto Moiré

Con effetto moiré si indica una figura di interferenza, creata ad esempio da due griglie uguali sovrapposte con diversa angolatura, o anche da griglie parallele con maglie distanziate in modo leggermente diverso.



Quantizzazione

Quando si misura una grandezza analogica, l'insieme di valori che essa può assumere in natura è un **insieme continuo** e composto da **infiniti punti**. A volte però nelle comunicazioni di tipo numerico o digitali il valore della grandezza in questione deve essere convertito in formato discreto. Ciò avviene preventivamente grazie ad un processo di campionamento.

Affinché una grandezza sia trasmisibile e codificabile con un numero finito di bit ovvero in forma numerica, è però necessario che essa possa assumere solo un **numero finito** di valori di codominio discreti; ciò avviene tramite un successivo processo di quantizzazione del valore in ordinata della grandezza in questione.

Per ottenere ciò i valori possibili della grandezza in questione vengono innanzitutto limitati tra un massimo ed un minimo intorno a dei valori discreti preventivamente definiti definendo così le relative regioni di decisione e la dinamica del quantizzatore stesso: in tal modo il valore analogico della grandezza originaria, in corrispondenza del valore campionato in ascissa, verrà ricondotto al più prossimo dei valori discreti preventivamente definiti tramite il processo di decisione.

Errori di quantizzazione

Con la quantizzazione vengono però introdotti degli errori detti errori di quantizzazione pari alla differenza tra il valore quantizzato e il suo valore "reale" nel campo continuo. L'errore massimo possibile che potrà essere introdotto volta per volta sarà quindi pari alla metà dell'intervallo discreto discriminabile o regione di decisione, nel caso limite in cui il valore di ingresso si collochi esattamente a metà tra due valori discreti di uscita ovvero sulla frontiera di due regioni di decisione contigue. L'insieme di questi errori conduce al **rumore di quantizzazione**. Nei CCD stessi a obiettivo chiuso ci sono correnti parassite che inducono rumore dentro il dispositivo elettronico dette "*dark current*".

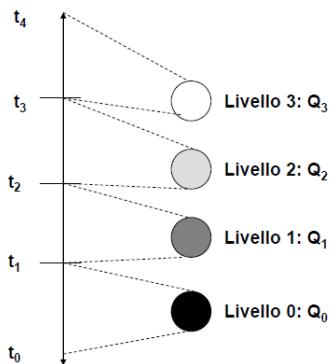
Il **Signal to Noise Quantization Ratio** (SNQR) misura la bontà del processo di quantizzazione ed è il parametro che più influisce sulla qualità del segnale digitalizzato.

Procedura generale

Se i valori da quantizzare sono numeri reali nel range $[a, b]$ e si vuole quantizzare su n livelli: Si fissano $n+1$ numeri in $[a, b]$:

$$t_0 = a < t_1 < t_2 < \dots < t_n < t_{n+1} = b$$

Il numero x in $[a, b]$ verrà assegnato al livello di quantizzazione k se risulta: $t_k \leq x < t_{k+1}$ (b viene assegnato al livello n)



Fissato il numero di livelli di quantizzazione si pone il problema di come rappresentare in memoria tali livelli. Ovviamente utilizzeremo delle etichette numeriche.

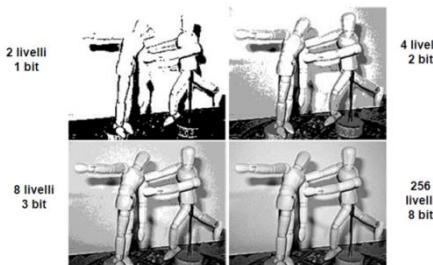
Quanti bit sono necessari per ricordare quale livello di luminosità si misura in un punto?

Nell'esempio ne bastano $2 = \log(4)$

In generale se ci sono N livelli occorre rappresentare N etichette numeriche e avremo bisogno di un numero di bit pari a:

$$B = \log(N)$$

Quantizzazione: effetti sulle immagini



La quantizzazione in particolare può essere di due tipologie: uniforme e non uniforme.

Quantizzazione uniforme

- range in ingresso $0 \dots N - 1$,
- range in uscita $0 \dots K - 1$ con $K \leq N$.

Se L è il livello di ingresso rappresentato da un intero il livello L' corrispondente dopo la quantizzazione è:

$$L' = \frac{L * K}{N}$$

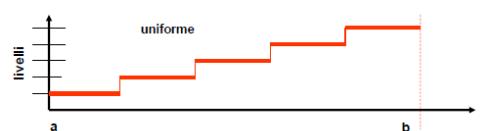
Es. Portare $0 \dots 255$ in $0 \dots 7$ con quantizzazione uniforme.

$$\text{Il livello 10 diviene } \frac{10*8}{256} = 0$$

$$\text{Il livello 20 diviene } \frac{20*8}{256} = 0$$

$$\text{Il livello 30 diviene } \frac{30*8}{256} = 0$$

$$\text{Il livello 32 diviene } \frac{32*8}{256} = 1 \text{ eccetera...}$$



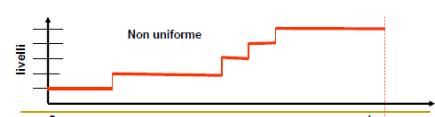
Quantizzazione non uniforme

- range in ingresso $0 \dots N - 1$,
- range in uscita $0 \dots K - 1$ con $K \leq N$

Se L è il livello di ingresso rappresentato da un intero il livello L' corrispondente dopo la riquantizzazione è:

$$L' = f(L, N, K)$$

La funzione $f(L, N, K)$ definisce lo schema di riquantizzazione, e può avere le forme più varie. Tra le più comuni è la quantizzazione logaritmica.



Quantizzazione logaritmica

$$f(L, N, K) = \frac{\log(L) * K}{\log(N)}$$

Nel caso più comune $N=256$, $\log(N)=8$ e $K=8$, o comunque in generale $\log(N)=K$, per cui $f(L,N,K)=\log(L)$. Tuttavia si potrebbe volere portare il range 0...255 in 0...15 e in tal caso la formula di cui sopra ritorna utile.



Es. $K = 3$

3	5	→	$\frac{\log(3) * 3}{8}$	$\frac{\log(5) * 3}{8}$
6	2		$\frac{\log(6) * 3}{8}$	$\frac{\log(2) * 3}{8}$

Quantizzazione nei dispositivi digitali

La quantizzazione effettuata dagli scanner commerciali e dalla fotocamere digitali è **non uniforme** e **logaritmica**: ciò permette di assegnare più livelli nella area dei toni scuri e meno livelli nella area dei toni chiari. Questo è particolarmente importante quando si elaborano dati medici (es.radiografie).

Risoluzione

Si dice risoluzione il numero di pixel per unità di misura. Essa si può misurare in pixel al centimetro, o in dots per inch (**dpi**); la misura può anche essere espressa come il numero di pixel su tutta l'immagine (es. 8 Megapixel, ovvero 8 milioni di pixel).

La risoluzione indica il grado di qualità di un'immagine. Lo schermo di un computer non può mostrare linee o disegni, ma soltanto punti; se questi sono sufficientemente piccoli, tali da essere più piccoli della risoluzione percepita dall'occhio umano, l'osservatore ha l'impressione di vedere linee anziché punti allineati, e disegni anziché ammassi di puntini distinti.

Il concetto di risoluzione è relativo e va considerato con attenzione a seconda di differenti situazioni:

- Risoluzione dell'apparecchiatura di ripresa

Si contano quanti sensori ci sono per unità lineare di misura.

SCANNER: fino a 6000 dpi e oltre

FOTOCAMERE: numero di sensori presenti sul circuito di ripresa. Si misura in MEGAPIXEL.

- Risoluzione dell'apparecchiatura di resa

Si contano quanti sensori ci sono per unità lineare di misura.

STAMPANTI: fino a 3000 dpi e oltre

SCHERMI: numero di elementi fluorescenti sullo schermo per unità di misura. Tipicamente 72 dpi.

- Risoluzione di stampe

Quotidiano: 75 dpi;

Riviste: 133 dpi;

Brochure: 175 dpi;

Libri fotografici: 2400 dpi.

Es. Un rosone a colori avente diametro di 2 metri.



L'immagine si compone di 200 pixel di larghezza.

Si ha una risoluzione "reale" di 1 dot per 1 cm.

Sono richiesti $200 \times 200 \times 24 \text{ bit} = 960\,000$ bit per la memorizzazione.

L'immagine si compone di 100 pixel di larghezza.

Si ha una risoluzione "reale" di 1 dot per 2 cm.

Sono richiesti $100 \times 100 \times 24 \text{ bit} = 240\,000$ bit per la memorizzazione.

L'immagine si compone di 10 pixel di larghezza.

Si ha una risoluzione "reale" di 1 dot per 20 cm.

Sono richiesti $10 \times 10 \times 24 \text{ bit} = 2400$ bit per la memorizzazione.

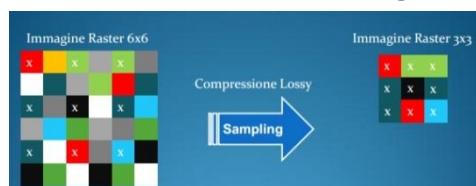
Sigla	Definizione	Risoluzione	Rapporto
Quarter QVGA	QQVGA	160 x 120	4:3
CGA o Quarter VGA	CGA / QVGA	320 x 240	4:3
Half VGA	HVGA	640 x 240	8:3
Monochrome Display Adapter	MDA	720 x 350	
Enhanced Graphics Array	EGA	640 x 350	
Video Graphics Array	VGA	640 x 480	4:3
Super VGA	SVGA	800 x 600	4:3
Quad VGA	QVGA	1280 x 960	4:3
eXtended Graphics Array	XGA	1024 x 768	4:3
Super XGA	SXGA	1280 x 1024	5:4
Super XGA Plus	SXGA+	1400 x 1050	4:3
Ultra XGA	UXGA	1600 x 1200	4:3
Quad XGA	QXGA	2048 x 1536	4:3
Quad Ultra XGA	QUXGA	3200 x 2400	4:3
(*) Wide XGA	WXGA	1366 x 768	~16:9
(*) Wide XGA	WXGA	1280 x 800	16:10
(*) Wide XGA	WXGA	1280 x 720	16:9
Wide XGA Plus	WXGA+	1440 x 900	16:10
Wide Super XGA Plus	WSXGA+	1680 x 1050	16:10
Wide Ultra XGA	WUXGA	1920 x 1200	16:10

Risoluzioni standard

Immagini nate con una certa risoluzione devono essere visualizzate con la stessa risoluzione per avere il massimo della resa.

Le principali dimensioni standard e i loro nomi sono riportati nella tabella a sinistra.

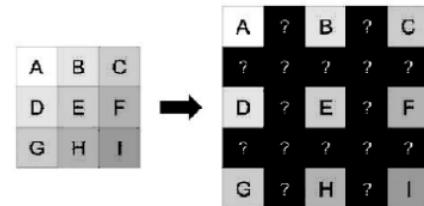
Ridimensionamento delle immagini



L'operazione di riduzione dell'immagine solitamente è la più facile e indolore. Essa consiste semplicemente nell'effettuare un'opportuna selezione (campionamento) dei pixel dell'immagine estesa, in modo da poter ricostruire con l'insieme dei campioni raccolti una versione ridotta dell'immagine. Quindi non vi è alcuna elaborazione da effettuare, i dati finali sono già noti a priori, si tratta solo di selezionarli.

L'operazione d'ingrandimento è ben più complessa della semplice riduzione; a differenza di quest'ultima ha una approfondita teoria matematica alle spalle.

In particolare, l'ingrandimento di un'immagine consiste nel effettuare una serie di operazioni al fine di poter "stimare" il valore da attribuire ai nuovi pixel. Tale stima non è univoca ma varia in base alla tecnica utilizzata, chiamata **interpolazione**.



Interpolazione

Il processo di interpolazione delle immagini digitali consiste fondamentalmente nel disporre i pixel dell'immagine originale su una matrice a più alta risoluzione (con un numero maggiore di righe e colonne) e di stimare il valore dei pixel mancanti utilizzando quelli noti.

In generale, l'interpolazione è il processo che partendo da dati reali stima i dati non conosciuti.

Naturalmente un'immagine interpolata non avrà mai la stessa qualità di un'immagine originale. I metodi di interpolazione, infatti, consistono nell'effettuare una media tra i pixel. Quindi ci si aspetta che un buon metodo di interpolazione si limiti a preservare le geometrie degli oggetti rappresentati nell'immagine (**edge**) senza introdurre artefatti.

Punto di vista matematico

In matematica per interpolazione si intende un metodo per individuare **nuovi punti** del piano cartesiano a partire da un insieme finito di punti dati, nell'ipotesi che tutti i punti si possano riferire ad una funzione $f(x)$ di una data famiglia di funzioni di una variabile reale.

Se si pensa ai pixel di un'immagine come ad una serie di punti in uno spazio tridimensionale del tipo riga, colonna, intensità del colore, si può individuare una funzione f interpolante tali punti

$$f : \text{riga} \times \text{colonna} \rightarrow \text{intensità del colore}$$

Grazie a questa funzione possiamo "conoscere" anche i valori dei pixel non noti a priori. Quindi le tecniche usate fanno uso dell'interpolazione.

Interpolazione unidimensionale

Sia data una sequenza di n numeri reali distinti x_k chiamati **nodi** e per ciascuno di questi x_k sia dato un secondo numero y_k . Ci proponiamo di individuare una funzione f di una certa famiglia tale che sia

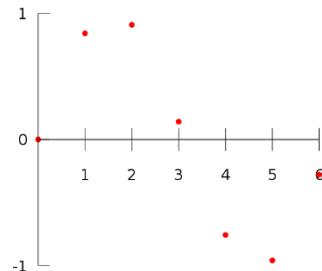
$$f(x_k) = y_k, \quad \text{per } k = 1, \dots, n$$

Una coppia (x_k, y_k) viene chiamato **punto dato** ed f viene detta **funzione interpolante** per i punti dati.

Si parla di interpolazione quando: note alcune coppie di dati $(x; y)$, interpretabili come punti di un piano, ci si propone di costruire una funzione, detta funzione interpolante, che sia in grado di descrivere la relazione che intercorre fra l'insieme dei valori x e l'insieme dei valori y .

Es. Si supponga di avere la seguente tabella, che dà alcuni valori di una funzione che si può considerare nota in altra sede.

x	f(x)
0	0
1	0.8415
2	0.9093
3	0.1411
4	-0.33
5	-0.9589
6	-0.2794



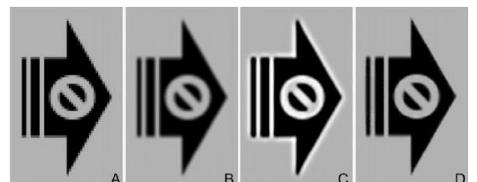
Ci chiediamo: quanto vale la funzione per esempio, in $x = 2.5$? L'interpolazione risolve problemi come questo.

Uno dei metodi più semplici è l'**interpolazione unidimensionale (lineare)**. Si consideri il suddetto esempio di determinare $f(2.5)$. Poiché 2.5 è il punto medio fra 2 e 3, è ragionevole assegnare a $f(2.5)$ come il valore tra la media di $f(2) = 0.9093$ e $f(3) = 0.1411$: in tal modo si ottiene $f(2.5) = 0.5252$.

Artefatti

Tipicamente i metodi di interpolazione introducono le quattro categorie di artefatti:

- A- **Aliasing**: le linee che definiscono gli edge appaiono frastagliate;
- B- **Blurring**: l'immagine appare sfocata;
- C- **Edge halo**: gli edge sono circondati da una banda luminosa;
- D- **Frequency artifact**: altri tipi di artefatti introdotti prevalentemente in prossimità di regioni ad alta frequenza.



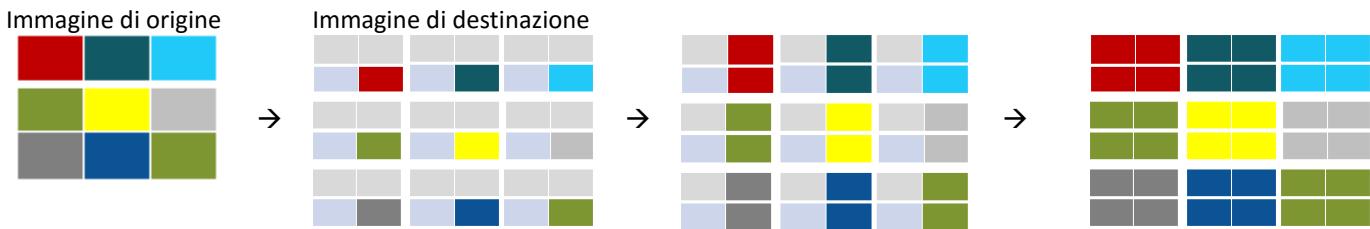
Vari tipi di interpolazione

Esistono diversi tipi di interpolazione:

- Nearest neighbor (o replication)
- Bilinear
- Bicubic
- Altri...

Nearest-neighbor (o Replication)

Letteralmente “Vicino più prossimo”: Questo metodo assegna a ogni nuova posizione dell’immagine di output l’intensità del pixel più prossimo nell’immagine originale. E’ sicuramente il metodo di interpolazione più semplice e veloce.



Questo approccio è molto semplice ma introduce artefatti come distorsioni lungo gli edge.

Esempio **Bayer Pattern di tipo GR/BG**

<table border="1"><tr><td>6</td><td>2</td><td>5</td></tr><tr><td>5</td><td>3</td><td>6</td></tr><tr><td>10</td><td>1</td><td>0</td></tr></table>	6	2	5	5	3	6	10	1	0	\rightarrow	<table border="1"><tr><td>G</td><td>R</td><td>G</td></tr><tr><td>B</td><td>G</td><td>B</td></tr><tr><td>G</td><td>R</td><td>G</td></tr></table>	G	R	G	B	G	B	G	R	G	\rightarrow	<table border="1"><tr><td>6</td><td>5</td><td>5</td></tr><tr><td>3</td><td>3</td><td>3</td></tr><tr><td>10</td><td>0</td><td>0</td></tr></table>	6	5	5	3	3	3	10	0	0	$\begin{matrix} G \\ R \\ B \end{matrix}$
6	2	5																														
5	3	6																														
10	1	0																														
G	R	G																														
B	G	B																														
G	R	G																														
6	5	5																														
3	3	3																														
10	0	0																														

Bilinear

Nell’interpolazione bilineare si utilizzano i quattro pixel più vicini per stimare l’intensità da assegnare a ciascuna nuova posizione. Supponiamo che (x, y) siano le coordinate della posizione cui si deve assegnare un valore di intensità e che $v(x, y)$ equivalga al valore dell’intensità. Per l’interpolazione bilineare il valore assegnato si ottiene mediante l’equazione

$$V(x, y) = ax + by + cxy + d$$

Dove i quattro coefficienti sono determinati a partire dalle quattro equazioni nelle quattro incognite ottenibili utilizzando i quattro pixel più vicini al punto (x, y) .

	0	1	2
0		5	
1	6	?	7
2		10	

$$V(x, y) = ax + by + cxy + d$$

$$\begin{cases} 5 = a0 + b1 + c01 + d \\ 6 = a1 + b0 + c10 + d \\ 7 = a1 + 2b + c21 + d \\ 10 = a2 + b1 + c21 + d \end{cases} \quad \begin{cases} 5 = b + d \\ 6 = a + d \\ 7 = a + 2b + 2c + d \\ 10 = 2a + b + 2c + d \end{cases} \quad \begin{cases} b = d - 5 \\ a = d - 6 \\ 7 = d + 2d - 10 + 2c + d - 6 \\ 10 = 2a + b + 2c + d \end{cases}$$

$$\begin{cases} b = d - 5 \\ a = d - 6 \\ c = \frac{-4d+23}{2} \\ 10 = 2a + b + 2c + d \end{cases} \quad \begin{cases} b = d - 5 \\ a = d - 6 \\ c = \frac{-4d+23}{2} \\ 10 = 2d - 12 + d - 5 - \frac{4d+23}{2} + d \end{cases} \quad \begin{cases} b = d - 5 \\ a = d - 6 \\ c = \frac{-4d+23}{2} \\ 10 = 4d - \frac{4d+23}{2} - 17 \end{cases}$$

$$\begin{cases} a = -6 \\ b = -5 \\ c = \frac{23}{2} \\ d = 0 \end{cases} \quad V(1,1) = -6 - 5 + \frac{23}{2} + 0 = \frac{1}{2} = 0 \quad (\text{arrotondato})$$

	0	1	2
0		5	
1	6	0	7
2		10	

L’interpolazione bilineare produce dei risultati migliori rispetto alla replication con un incremento modesto nella complessità di calcolo.

Bicubic

L’interpolazione bicubica utilizza i sedici pixel più vicini al punto. Il valore di intensità assegnato al punto (x, y) si ottiene attraverso l’equazione

$$v(x, y) = \sum_{i=0}^3 \sum_{j=0}^3 a_{ij} x^i y^j$$

Dove i sedici coefficienti sono determinati a partire da sedici equazioni in sedici incognite che possono essere scritte utilizzando i sedici punti più vicini a (x, y) .

Generalmente l’interpolazione bicubica preserva meglio i dettagli rispetto all’interpolazione bilineare. L’interpolazione bicubica è la tecnica standard utilizzata nei programmi commerciali di editing come Adobe Photoshop e Corel Photopaint.

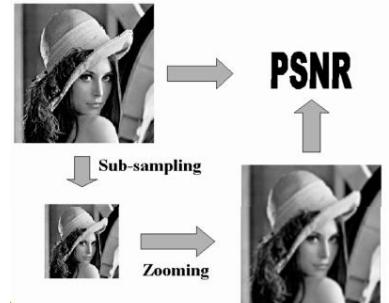
ATTENZIONE: L’interpolazione non comporta un miglioramento della qualità dell’immagine come se “riacquisisse” i valori mancanti ma effettua solo una stima dei valori ignoti.

Valutazione degli algoritmi di interpolazione

Per valutare un buon algoritmo di interpolazione solitamente si usa calcolare il **PSNR** (Peak Signal to Noise Ratio), è una misura adottata per valutare la qualità di una immagine compressa rispetto all'originale. Per calcolarlo è necessario avere sia l'immagine interpolata che una sua versione originaria.

È più facile da definire attraverso l'errore quadratico medio (MSE).

$$MSE = \frac{1}{MN} \sum_{x=1}^M \sum_{y=1}^N [I'(x, y) - I(x, y)]^2$$



dove M è il numero di righe della matrice delle immagini, N le colonne, I l'immagine originale, I' l'immagine interpolata ed S è il massimo picco dei valori dei pixel (di solito 255)

$$PSNR = -10 \log_{10} \frac{MSE}{S^2} \quad PSNR = 20 \log_{10} \left(\frac{S}{\sqrt{MSE}} \right) \quad PSNR = 10 \log_{10} \left(\frac{S^2}{MSE} \right)$$

Il PSNR non è il migliore parametro per valutare la qualità di un algoritmo di interpolazione, ma è il più diffuso.
Altre misure più attendibili si basano sulla qualità percettiva dell'immagine.

Es. Calcolare il PSNR tra l'immagine e la sua trasposta

6	2	5
5	3	6
10	2	0

|

10	5	6
2	3	2
0	6	5

|'

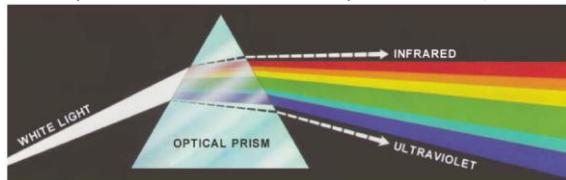
$$\begin{aligned} MSE &= \frac{1}{9} \cdot [(10-6)^2 + (5-2)^2 + (6-5)^2 + (2-5)^2 + (3-3)^2 + (2-6)^2 + \\ &\quad (0-10)^2 + (6-2)^2 + (5-0)^2] = \\ &= \frac{1}{9} \cdot [16 + 9 + 1 + 9 + 0 + 16 + 100 + 16 + 25] = \frac{1}{9} \cdot 192 = \frac{192}{9} = \frac{64}{3} \\ PSNR &= -10 \log_{10} \frac{\frac{64}{3}}{(10)^2} = -10 \log_{10} \frac{64}{3} \cdot 100 = 6.709412807357753 \end{aligned}$$

Colore

Abbiamo studiato come funziona l'occhio e come il cervello elabora le informazioni ricevute. Adesso dobbiamo capire come è fatta la luce!

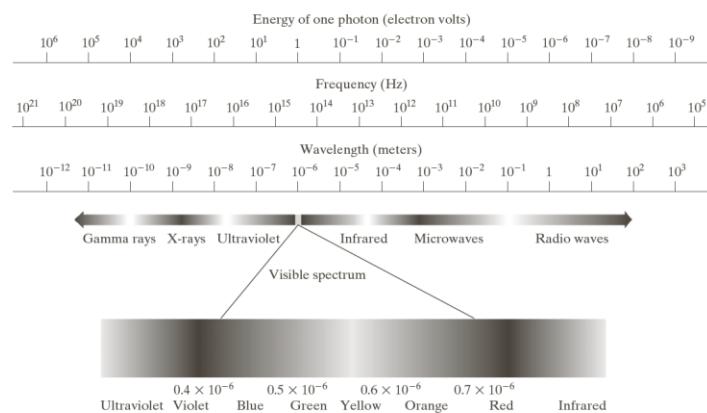
Luce

Se un raggio luminoso bianco attraversa un prisma di vetro, ciò che si ottiene non è luce bianca, ma è uno spettro di colori che vanno dal violetto al rosso. Quindi la luce può essere decomposta in onde luminose di tipo differente (Sir Isaac Newton, 1666).

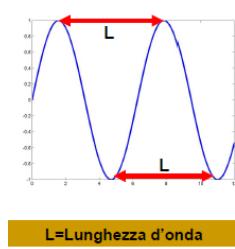


Spettro elettromagnetico

Quello che il nostro occhio percepisce è solo una piccola porzione dello spettro elettromagnetico.



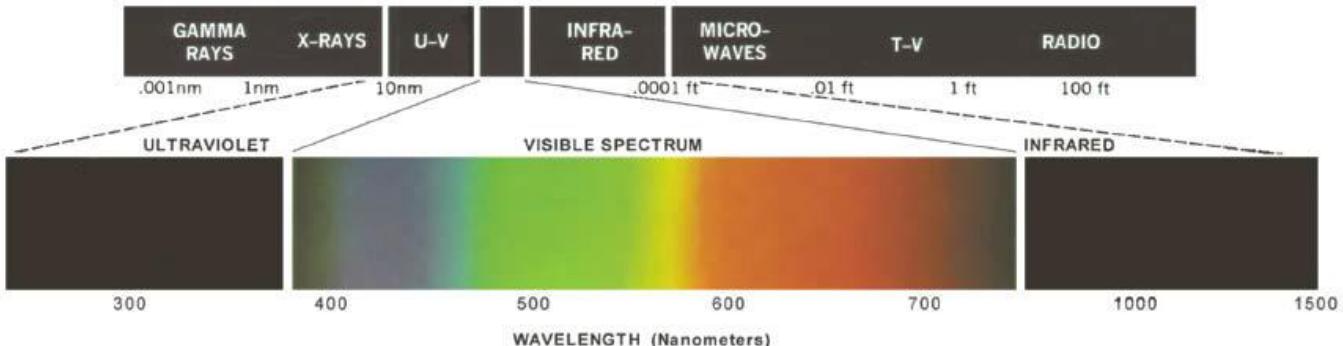
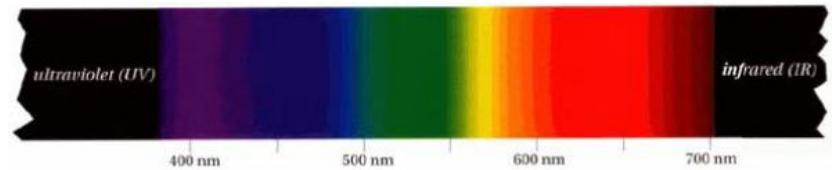
Lunghezze d'onda



Lung. in nanometri	Tipo radiazione
$10^{17} - 10^{13}$	Osc.elettriche
$10^{13} - 10^9$	Onde radio
$10^9 - 10^6$	Micro-onde
$10^6 - 10^3$	Infrarosso
$10^3 - 10^2$	Visible
$10^2 - 10$	Ultravioletto
$10 - 10^{-3}$	Raggi X
$10^{-3} - 10^{-7}$	Raggi gamma e cosmici

Un nanometro = 1 metro / 1.000.000.000

I colori sono legati alle lunghezze d'onda (Newton-Huygens). La luce è formata da tutte quelle lunghezze d'onda percepite dall'occhio umano. Lo spettro della luce visibile oscilla tra il violetto e il rosso. Per comodità lo spettro del visibile è diviso in sei regioni: violetto, blu, verde, giallo, arancio e rosso. Le bande di colore non sono tutte della stessa grandezza e degradano in quelle limitrofe.



Percezione dei colori

L'occhio umano percepisce come colore di un oggetto quella luce che l'oggetto stesso riflette. Se un oggetto riflette tutte le lunghezze d'onda luminosa, allora l'oggetto sarà percepito come bianco. Un oggetto che riflette le lunghezze d'onda da 500 a 570 nm ed assorbe tutto il resto, sarà percepito come di colore verde.

Per descrivere la luce bastano i seguenti valori:

- **Radianza:** cioè la quantità di luce emessa dalla sorgente luminosa;
- **Luminanza:** cioè la misura dell'energia percepita dall'utente;
- **Brillantezza:** è un valore soggettivo che indica la sensazione di colore.

I Coni

Nella retina ci sono tre tipi di coni:

- **TIPO S:** Sensibili alle lunghezze d'onda corte (short, colori bluastri)
- **TIPO M:** Sensibili alle lunghezze d'onda medie (middle, colori verdastri)
- **TIPO L:** Sensibili alle lunghezze d'onda lunghe (long, colori rossastri)

Tutti i primati hanno questi tre tipi di cellule retinali. I non primati hanno solo due tipi di cellule retinali per i colori mentre gli uccelli ne hanno ben 5 tipi differenti.

Standard dei Colori

Teoria del tristimolo (Young, 1802)

Tutti i colori si possono ottenere “mescolando” tre colori fondamentali in proporzioni differenti.

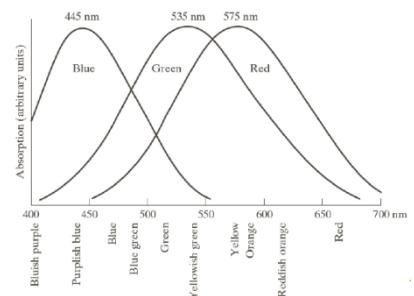
Dimostreremo che questa *ipotesi* è *FALSA* se non in prima approssimazione.

Standard CIE

Nel 1931 il CIE (Commission Internationale de l'Eclairage) ha fissato le lunghezze d'onda standard per i tre colori primari:

- Blu = 435,8 nm
- Verde = 546,1 nm
- Rosso = 700 nm

Anche se nel 1965 i dati sperimentali hanno dimostrato che in realtà il valore reale è lievemente differente.

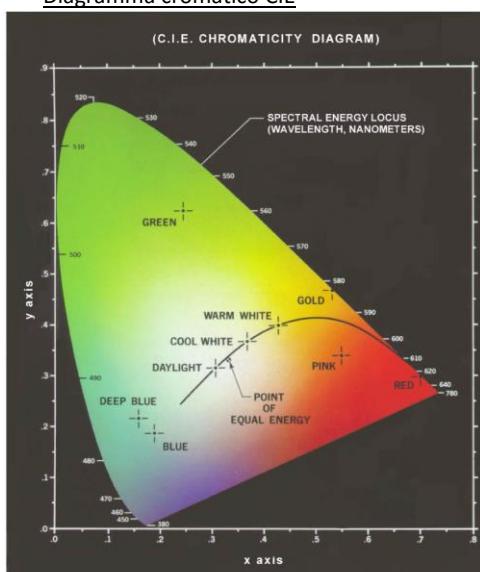


Colori primari e secondari

I colori **Rosso (R)**, **Verde (G)** e **Blu (B)** sono detti **colori primari**. Combinandoli tra di loro NON si ottengono tutti i colori visibili. Combinandoli a due a due si ottengono i **colori secondari**: **Magenta (M)**, **Giallo (Y)** e **Ciano (C)**.

L'uso del termine primario è stato ampiamente frainteso nel senso che i tre colori primari standard, mescolati in varie proporzioni di intensità, venivano considerati capaci di produrre tutti i colori visibili. Come si vedrà a breve, questa interpretazione non è corretta a meno che si permetta anche alla lunghezza d'onda di variare, ma in questo caso non si avrebbero tre colori primari standard fissi.

Diagramma cromatico CIE



Dati x = quantità di rosso, y = quantità di verde, z = quantità di blu ottenuta come $z = 1 - (x + y)$.

La rappresentazione grafica al variare di x e y da origine al diagramma cromatico CIE.

Tutti i colori delle lunghezze d'onda visibili sono disposti lungo i bordi.

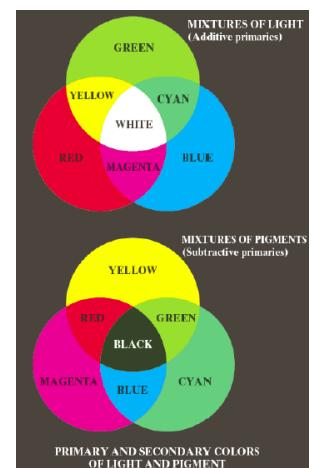
Il punto di uguale energia è il bianco.

Qualsiasi colore lungo il bordo non ha bianco, quindi è puro.

Unendo due colori con una linea, tutti i colori nella linea sono quelli ottenibili mischiando i due colori.

Unendo un colore con il bianco si ottengono tutte le tonalità di quel colore.

Unendo tre colori con un triangolo, tutti i colori lungo il bordo e nel triangolo sono quelli ottenibili mischiando quei tre colori.

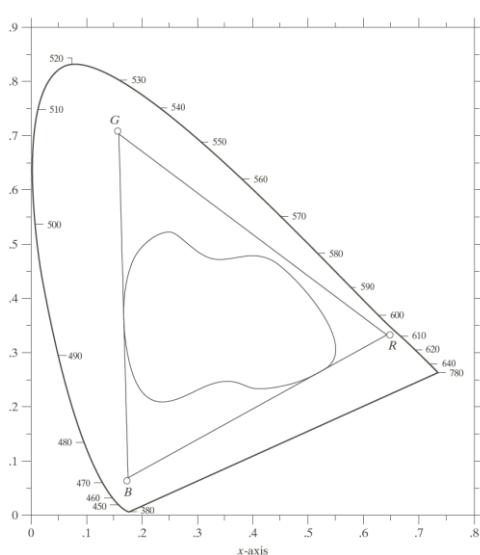


Unendo R G e B si ottiene un triangolo che contiene tutti i colori che si possono produrre.

Da notare che il triangolo non copre tutta l'area, quindi non tutti i colori si ottengono unendo R G e B.

L'area irregolare dentro il triangolo rappresenta tutti i colori che una stampante può ottenere.

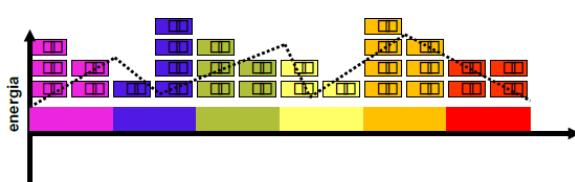
Questi sono in numero minore rispetto a quelli dei monitor perché è differente il modo di mischiare i colori (additivo vs sottrattivo).



Gli spazi di colore

In realtà raramente vediamo in natura colori puri. Ma piuttosto vediamo miscele di radiazione luminosa in ogni lunghezza d'onda.

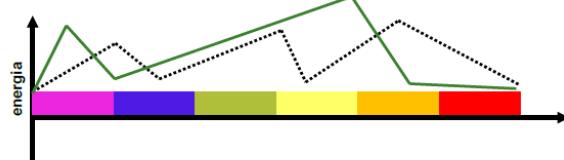
Il nostro cervello **non è uno spettrometro**: spettri differenti producono sensazioni cromatiche eguali; manteniamo una percezione costante del colore di una superficie anche se cambia la luce che la illumina.



Ogni lunghezza d'onda trasporta differenti quantità di energia. L'energia totale della radiazione è la somma di tutti i contributi di energia dalle diverse lunghezze d'onda. Lo **spettro di un illuminante** è il diagramma dei contributi di energia che esso apporta per ciascuna differente lunghezza d'onda.

Il disegno mostra uno spettro "discretizzato" con una convenzione grafica del tipo visto in una applicazione di "equalizzazione" nello stereo.

Metameri

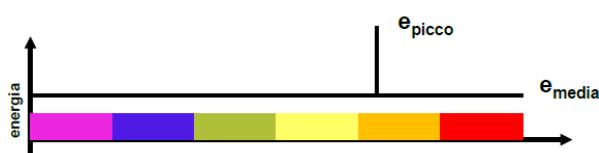


Spettri diversi possono produrre colori uguali: coppie di spettri con questa reciproca proprietà si chiamano **metameri**.

Tra i vari metameri di un dato spettro se ne può sempre individuare uno assai importante che è alla base del modello dei colori detto "**del pittore**".

Lo spettro tratteggiato e quello continuo producono (nel cervello) il medesimo colore!

Modello del pittore



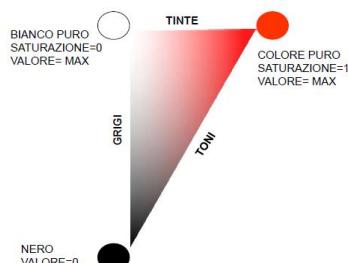
Ogni spettro ha un metamero della forma a sinistra.

La lunghezza d'onda in cui si ha il picco è responsabile del **colore percepito** (detto anche in inglese HUE).

Il rapporto $(e_{\text{picco}} - e_{\text{media}}) / (e_{\text{picco}} + e_{\text{media}})$ è la **SATURAZIONE**, cioè quanto il colore è puro. Meno luce bianca equivale ad un maggiore valore del rapporto.

e_{media} è proporzionale al contenuto energetico della radiazione: essa può essere considerata una misura della **luminosità** di una radiazione (detto anche **VALORE**). Esso dà un contributo "bianco" al colore percepito.

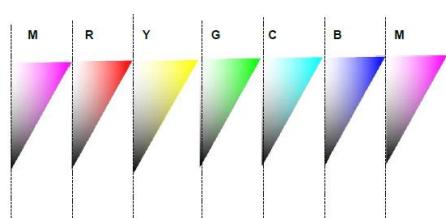
Modello del pittore: schema di Munsell



Questo è il modello usato nel mondo dell'arte e insegnato nelle accademie.

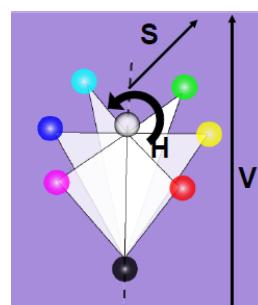
Si ha un triangolo come questo per ciascuna HUE differente

Spazio HVS (oppure HSI)

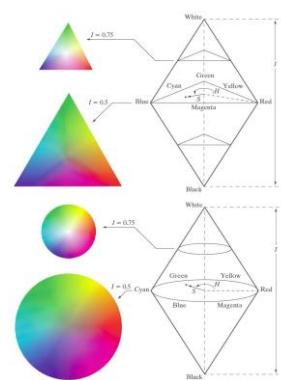
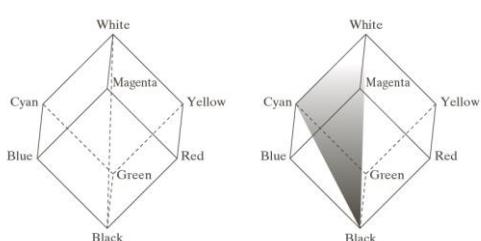


Tutte le linee verticali tratteggiate sono la rappresentazione della medesima "linea dei grigi". IDEA: Attacchiamo tutti i triangoli in una "girandola" facendo coincidere le linee dei grigi. Troveremo la figura a destra, avente:

- **H = hue (colore):** copre tutti i colori ordinati in sequenza
- **S = saturazione:** da un minimo (centro) pari al bianco puro ad un massimo (periferia) colore puro.
- **V = valore o luminosità:** da un minimo (nessuna energia emessa) ad un massimo.



Relazione tra RGB e HSV



Pro e contro del modello del pittore

Pro

- Intuitivo;
- Percettivamente significativo: i parametri HSV hanno una perfetta interpretazione nelle nostre percezioni.

Contro

- Modello non lineare;
- Perché una piramide esagonale?
- Quanti sono i "colori base"?

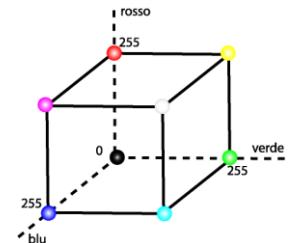
Modello RGB



Nel modello RGB ciascun colore è ottenuto mischiando i tre colori fondamentali. Se ogni componente di colore è intesa come una coordinata cartesiana, allora il modello RGB può essere graficamente descritto da un cubo.

I contributi del RED, GREEN e BLUE sono assunti indipendenti l'uno dall'altro (e quindi rappresentati da direzioni perpendicolari tra loro). Ogni colore è un punto contenuto dentro il cubo. La retta che congiunge nero e bianco è la retta dei grigi.

Nel mondo delle immagini a colori ci sono due modi per riprodurre selettivamente la cromaticità (specificazione oggettiva della qualità di un colore indifferentemente dalla sua luminanza): **colori additivi** e **colori sottrattivi**.



Modello additivo

I colori additivi sono utilizzati con sistemi che emettono luce, nei quali la luce proveniente da sorgenti di differente colore viene fusa assieme per produrre i colori così come sono percepiti.

In un dispositivo a colori additivi, come un *display CRT*, la luce è prodotta da tre fosfori primari, rosso, verde, e blu (RGB). Questi fosfori vengono eccitati separatamente formando un fascio di elettroni. La luce emessa dai tre fosfori stimola nell'occhio i tre tipi di recettori per produrre la percezione dei colori.

Pro

- Semplice da usare e implementare in software e hardware.
Di fatto è uno STANDARD.

Contro

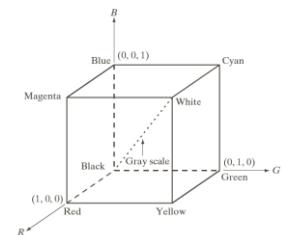
- percettivamente poco comodo: difficile capire guardando un colore in natura in quale proporzione vi contribuiscono l'R, il G e il B.

Il CUBO RGB e la piramide HSV si trasformano l'un l'altro mediante semplici algoritmi (non lineari).

Modello sottrattivo (CMY)

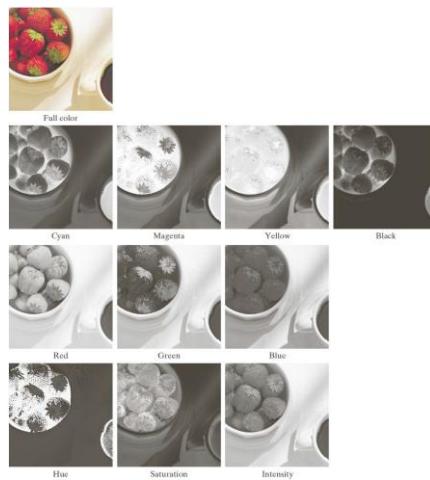
Il modello complementare a RGB è CMY, Ciano, Magenta e Giallo.

I colori sottrattivi sono usati nei sistemi passivi, nei quali la luce viene assorbita selettivamente alle diverse lunghezze d'onda, riflettendone solo alcune che comporranno i colori desiderati.



I colori sottrattivi sono utilizzati principalmente nell'industria della *stamp*a. Tre colori (e a volte un quarto) vengono impressi in una superficie riflettente come la carta. Gli inchiostri, tipicamente ciano (un blu-verde), magenta, e giallo (CMY), assorbono selettivamente una certa gamma di lunghezze d'onda della luce. L'occhio percepisce la luce riflessa, che non viene assorbita; da qui il termine "sottrattivi".

Quando non c'è inchiostro nella carta la luce riflessa è bianca; quando sono presenti tutti e tre i colori, la luce viene (in linea di principio) assorbita e la carta appare nera. In pratica, l'assorbimento completo è difficile da ottenere quindi si utilizza un quarto inchiostro, il nero (CMYK, dove K sta per black).



Riepilogando

- Le rappresentazioni dei colori nello spazio *RGB* (o *CMY* il duale) non sempre sono le più convenienti.
- Sono disponibili altre rappresentazioni che usano componenti che sono specificatamente relazionate al criterio usato per descrivere la luminanza, la tinta e la saturazione (*HSV*).
- La *tinta* descrive che colore è presente (rosso, verde, giallo, ecc.) e può essere correlato alla lunghezza d'onda dominante della sorgente di luce.
- La *saturazione*, invece, esprime quanto è vivo il colore (molto forte, pastello, vicino al bianco) e può essere correlato alla purezza o alla distribuzione dello spettro della sorgente.

- La **luminanza** è la grandezza che tende a valutare la sensazione luminosa ricevuta dall'occhio, è legata quindi all'intensità della luce (quanto il colore è bianco, grigio o nero) e può essere correlata alla luminosità della sorgente.

Colori sicuri

Si sa che 40 di questi 256 colori vengono processati in modo diverso da vari tipi di sistemi operativi, mentre **216 colori sono comuni** alla maggior parte dei sistemi. Questi 216 colori sono diventati gli standard di fatto dei colori sicuri, specialmente nelle applicazioni Internet. Essi vengono utilizzati quando si vuole che i colori visti dalla maggior parte delle persone siano gli stessi.

Ognuno dei 216 colori sicuri è formato come sempre da tre valori RGB, ma ogni valore può essere solo 0, 51, 102, 153, 204 o 255. Quindi, le triplette RGB di questi valori ci danno $(6)^3 = 216$ valori possibili (si noti che tutti i valori sono divisibili per 3).

Rappresentazioni luminanza-crominanza

Gli spazi colore, nei quali una componente è la luminosità e le altre due componenti sono legate alla tinta e alla saturazione, vengono chiamate rappresentazioni luminanza-crominanza.

La luminanza fornisce una versione a scala di grigi dell'immagine mentre la crominanza fornisce le informazioni "extra" che trasformano l'immagine in scala di grigi in un'immagine a colori.

Le rappresentazioni luminanza-crominanza sono particolarmente importanti nella compressione delle immagini.

L'occhio umano è più sensibile alla luminanza che ai colori. Posso dunque "spendere" molti bit per registrare la luminanza e risparmiarne un po' sulle crominanze.

Spazio YUV

La **luminanza** è ottenuta mediante una combinazione lineare della intensità luminosa dei canali rosso, verde e blu. Un'approssimazione abbastanza fedele della luminanza Y si ottiene attraverso la somma pesata:

$$Y = 0.3R + 0.6G + 0.1B$$

Il termine **crominanza** è definito come la differenza tra il colore e un bianco di riferimento alla stessa luminanza. I valori della crominanza possono pertanto essere espresse da un insieme di differenze di colore, U e V , definiti come segue:

$$\begin{aligned} U &= B - Y & U \in [-229.5, 229.5] \\ V &= R - Y & V \in [-178.5, 178.5] \end{aligned}$$

Queste differenze di colore valgono 0 quando $R=G=B$. Questa condizione produce il grigio che non ha crominanza. La componente V controlla i colori dal rosso al blu-verde. Mentre la componente U controlla i colori dal blu al giallo.

Luminanza



$$L = aR + bG + cB$$

$$\begin{aligned} a &= .3 \\ b &= .6 \\ c &= .1 \end{aligned}$$



I tre canali RGB non danno eguale contributo alla luminanza.
Il valore di luminanza è mantenuto massimamente nel canale G.

Immagine RGB

Canale LUMINANZA

Spazio YC_bC_r

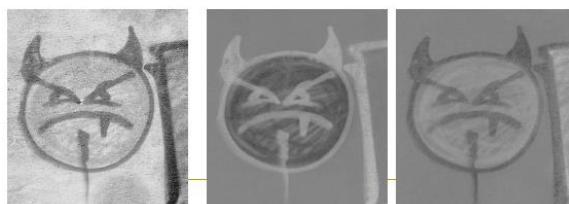
Lo spazio YC_bC_r è fortemente legato allo spazio YUV. Esso utilizza la stessa coordinata del sistema YUV, mentre le coordinate U e V vengono scalate e shiftate producendo due variabili, C_b e C_r rispettivamente. In particolare le equazioni per C_b e C_r sono:

$$C_b = \frac{U}{2} + 114,75 \quad C_r = \frac{V}{1.6} + 111,57$$

Con questa trasformazione i valori della crominanza stanno sempre nell'intervallo da 0 a 255. Questo sistema di coordinate viene largamente utilizzato dagli standard di compressione (Es. JPEG).

YC_bC_r :

Y rappresenta la *luminanza* mentre C_b e C_r rappresentano la *crominanza* del blu e del rosso.

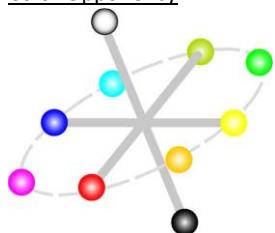


Minima differenza percettiva

Se si variano i parametri (RGB oppure HSV) di pochissimo il colore percepito resta eguale all'occhio di un umano.

La minima differenza percettiva è il valore max entro il quale contenere le variazioni dei parametri per non variare percettivamente il colore. In RGB e HSV non esiste una unica "minima differenza percettiva" sull'intero spazio (colori più saturi sono più sensibili alle variazioni dei parametri di quelli meno saturi). I modelli RGB e HSV NON sono percettivamente uniformi. I modelli luminanza-crominanza SONO percettivamente uniformi.

Color Opponency



Due coppie di colori sono "opposte" (ovvero danno massimo contrasto): BLU-GIALLO VERDE-ROSSO.

Occorre usare uno sfondo blu e scritte gialle per il massimo contrasto tra colori (nero su bianco darà comunque maggiore contrasto!).

Se si vuole "estrarre" una regione da una immagine tratta dalla natura basandosi sul suo colore, guardare ai valori RGB porta a risultati poco soddisfacenti. Se si guardano i valori nello spazio Color Opponency invece si ha una migliore "estrazione".

Colori e memoria

Lo schema assunto in RAM per mostrare i colori è:

$$8 \text{ bit Red} + 8 \text{ bit Green} + 8 \text{ bit Blue} = 24 \text{ bit}$$

(circa 16 milioni di colori o true color)

Questo costoso schema NON E' lo schema con il quale i colori vengono conservati in memoria di massa e compressi nelle tecniche JPEG, GIF o altro. Un'immagine ha più colori che pixel. Ad esempio un'immagine piccola è di $800 \times 600 = 480.000$ pixel. Inoltre le immagini "naturali" hanno una proprietà di **coerenza interna** per cui raramente si ha un colore differente per ogni differente pixel.

Questo porta ad adottare la modalità a **COLORI INDICIZZATI** (indexed color) o a **PALETTE** o a **LOOK-UP-TABLE** (LUT).

Palette



Dovrei ricordare:

255, 0, 0	255, 0, 0	0, 255, 0
255, 0, 0	255, 0, 0	0, 255, 0
0, 0, 255	255, 255, 255	255, 0, 0

Totale (9 pixel x 3 byte)= 27 byte = 216 bit

00	00	01	00 = (255, 0, 0)
00	00	01	01 = (0, 255, 0)
11	10	00	10 = (255, 255, 255)
11	10	00	11 = (0, 0, 255)

Totale
18 bit (9 pixel x 2
bit) per l'immagine
+ 12 byte per la
paletta
= 114 bit

Ricorda queste "etichette" e
questa tabella

La tabella che lega "etichette" con le corrispondenti componenti RGB si chiama:
"tavoloza",
"palette",
"tabella di indicizzazione dei colori",
"tabella di sbirciata",
look up table",
"LUT".

Indicizzazione di una immagine true color

I software commerciali e alcuni formati di compressione (**GIF**) adottano una paletta di 256 colori. Se nell'immagine true color ci sono meno di 256 colori essi vengono replicati nella paletta. Se nell'immagine true color ci sono più di 256 colori essi vengono "ridotti" scegliendo 256 rappresentanti che garantiscono una buona qualità visiva (esistono numerosi algoritmi, anche proprietari per tale scopo). Esistono anche palette "standard": MAC, WINDOWS, WEB_SAFE, OTTIMIZZATE eccetera.



Paletta a 256 colori

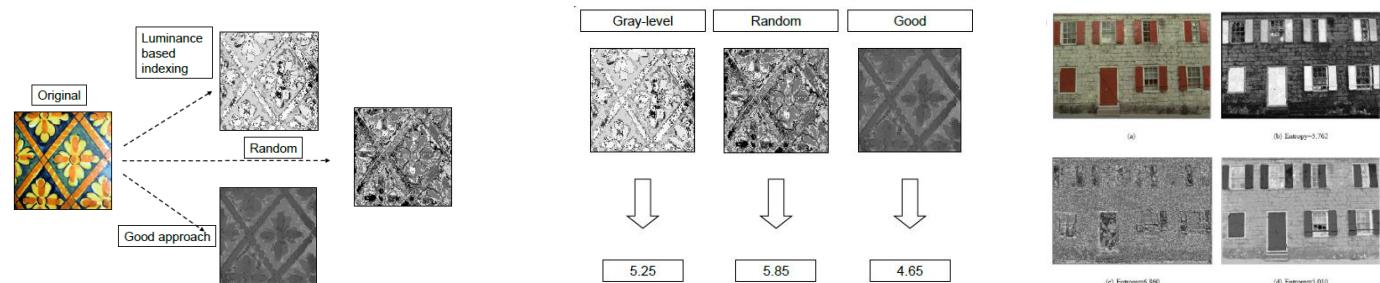
Paletta a 16 colori

Paletta a 8 colori

Paletta a 4 colori

Re-indexing

Re-indirizzare significa cambiare la posizione della paletta e quindi anche l'indice corrispondente in modo da creare una matrice di indici che abbia l'entropia minima.



Operazioni Puntuale

Per semplificare la trattazione dei problemi con le operazioni puntuali lavoreremo solo su immagini a toni di grigio. Le medesime operazioni descritte per tali immagini si estendono alle immagini RGB operando separatamente sui tre canali (piani) R, G e B e trattando ciascuno di essi come una immagine a toni di grigio indipendente dagli altri canali .

L'istogramma

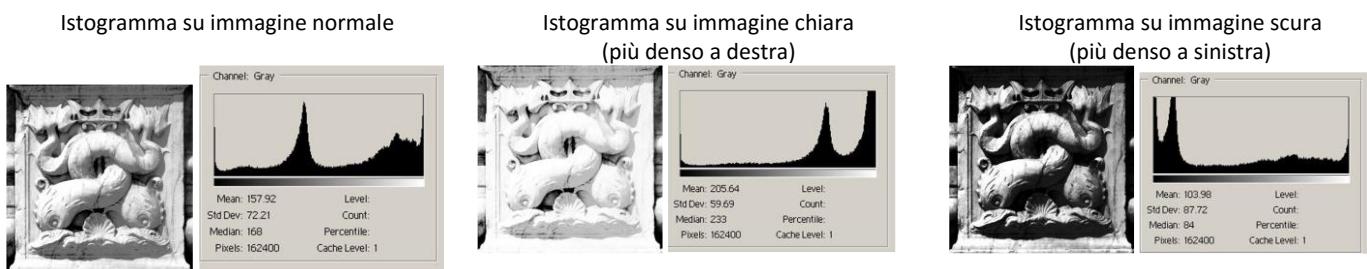
I pixel di una immagine sono una “popolazione” sulla quale possiamo calcolare tutte le quantità statistiche descrittive che si usano normalmente: Media, mediana, varianza, deviazione standard, quartili, percentili ...

Particolarmente importante è la conoscenza della **distribuzione delle frequenze dei toni di grigio**: l'istogramma.

L'istogramma, dunque, per ogni livello di grigio riporta il numero di pixel di quel colore. Per una immagine $I[m, n]$ si ha

$$H(k) = \text{numero di pixel di valore } k$$

E la somma di tutti gli H è esattamente $m \times n$. L'istogramma è utile per comprendere in maniera immediata le caratteristiche dell'immagine.

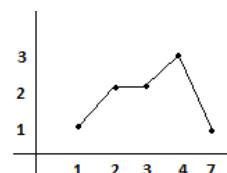


Inoltre immagini diverse potrebbero avere istogrammi simili: l'istogramma non tiene conto della **distribuzione spaziale** dei pixel.

Es. Disegnare l'istogramma dell'immagine avente matrice

$$\begin{bmatrix} 2 & 3 & 4 \\ 7 & 2 & 3 \\ 4 & 4 & 1 \end{bmatrix}$$

Costruiamo il grafico assegnando all'asse X il valore dei pixel della matrice in ordine crescente, e all'asse Y il numero di ripetizioni del valore del pixel come nella figura a destra



Dominio del tempo e dominio delle frequenze

Il termine Dominio del tempo (o dello spazio) è usato per descrivere l'analisi di funzioni matematiche, o di segnali rispetto alla variabile tempo.

Il termine Dominio delle frequenze, invece, viene usato nel contesto dello studio delle funzioni matematiche e dei segnali quando tali entità sono descritte mediante l'analisi dello spettro delle frequenze costitutive piuttosto che mediante il loro andamento nel tempo

Un grafico nel dominio del tempo mostra come un segnale cambia nel tempo, mentre un grafico nel dominio della frequenza mostra come e quanto un segnale si suddivide o è distribuito nelle varie bande di frequenza, definite all'interno di un dato range.

Tipi di operazioni sulle immagini

Le operazioni applicate alle immagini alterano i valori dei pixel di una immagine. L'immagine finale apparirà differente da quella iniziale. Questi operatori lavorano sia su immagini a colori che su immagini a toni di grigio.

Le elaborazioni nel dominio spaziale possono essere espresse come:

$$g(x, y) = T[f(x, y)]$$

essendo f l'immagine di ingresso alla elaborazione, g quella di uscita e T un operatore su f definito in un intorno di (x,y).

La dimensione dell'intorno di (x,y) definisce il carattere della elaborazione:

- puntuale (l'intorno coincide con il pixel stesso);
- locale (per esempio una piccola regione quadrata centrata sul pixel);
- globale (l'intorno coincide con l'intera f).

Operatori puntuali

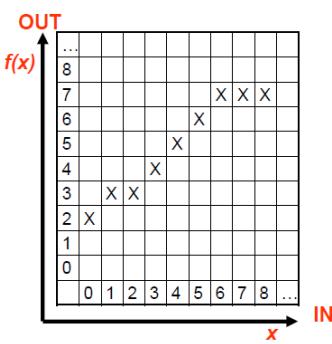
Si dice operatore puntuale, un operatore che preso in input il valore di un pixel ne restituisce uno cambiato che dipende esclusivamente dal valore del pixel in ingresso. Tipiche operazioni puntuali:

- aggiunta o sottrazione di una costante a tutti i pixel (per compensare sotto o sovraesposizioni);
- inversione della scala dei grigi (negativo);
- espansione del contrasto;
- modifica (equalizzazione o specifica) dell'istogramma;
- presentazione in falsi colori.

Un operatore puntuale può essere rappresentato da una funzione che preso in input un valore x lo modifica in un valore $y = f(x)$ con x,y appartenenti allo stesso campo di definizione (es. entrambi tra 0 e 255).

Poiché un operatore puntuale dipende solo dal valore del pixel esso è completamente descritto da una tabella come quella a destra:

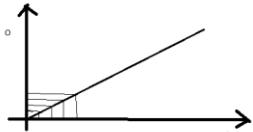
IN	0	1	2	3	4	5	6	7	...
OUT	$f(0)$	$f(1)$	$f(2)$	$f(3)$	$f(4)$	$f(5)$	$f(6)$	$f(7)$...



Questa a sinistra è universalmente l'interfaccia che tutti i programmi commerciali di immagini offrono per la visualizzazione e gestione delle operazioni puntuale

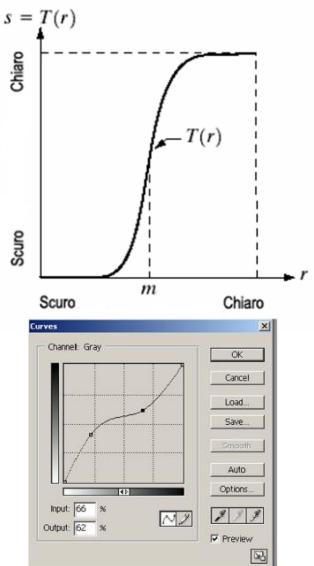
LUT

Le operazioni puntuali sono definite tramite sistema di riferimento cartesiano, con nell'asse delle x i valori in input e nell'asse delle y l'output (grafico LUT, *look-up tables*, grafico usato per definire le operazioni puntuale) come nella figura in alto a destra.



In Photoshop si chiama "aggiusta curve" (figura a destra).

A sinistra esempio di operazione di identità con il grafico LUT



Negativo

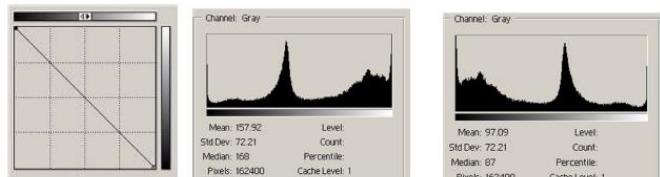
È la più semplice operazione puntuale. Consiste nell'associare al valore x del pixel il valore $255-x$.



Come cambia la curva



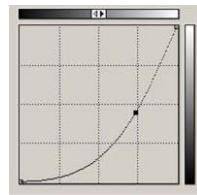
Come cambia l'istogramma



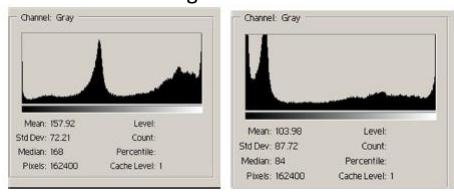
Incupimento dell'immagine



Come cambia la curva



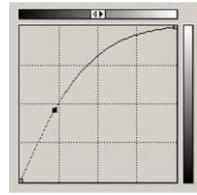
Come cambia l'istogramma



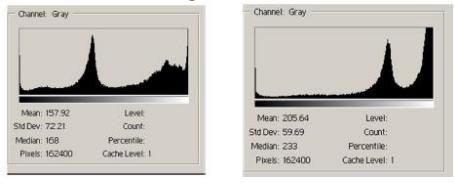
Schiarimento dell'immagine



Come cambia la curva



Come cambia l'istogramma



Trasformazione logaritmica

Si tratta di una trasformazione che consente di comprimere la gamma dinamica, permettendo la memorizzazione o la visualizzazione, con una scala dei grigi usuale, di immagini caratterizzate da escursioni di intensità molto ampie (molto scure). Può essere espressa come

$$s = c \log(1 + r)$$

Dove c è una costante positiva che serve a normalizzare il risultato tra 0 e 255.

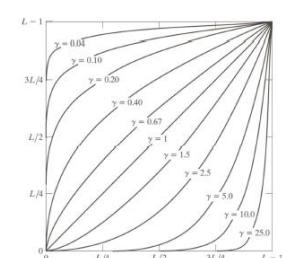
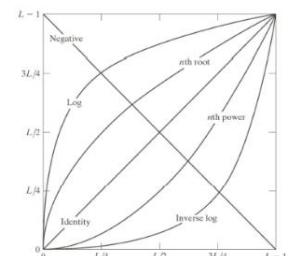
Trasformazione di potenza

La trasformazione di potenza può essere espressa come

$$s = cr^\gamma$$

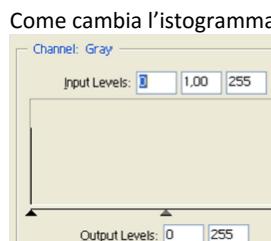
dove c e γ (gamma) sono costanti positive. La costante c è scelta di volta in volta in modo da normalizzare i valori di s nell'intervallo [0, 255]. Come vedremo, per valori di γ minori di 1 la trasformazione ha effetti analoghi alla trasformazione logaritmica (espansione della dinamica per bassi valori di r, compressione della dinamica per alti valori di r), mentre per valori di γ maggiori di 1 la trasformazione ha esattamente gli effetti opposti.

Su un monitor CRT (con $\gamma = 2.5$) si può applicare una correzione pre-processando l'input con la corrispondente funzione inversa: $s = r^{1/2.5} = r^{0.4}$



Binarizzazione

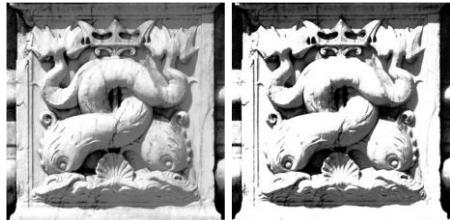
Produce una immagine che ha solo due livelli: nero e bianco. Si ottiene scegliendo una soglia T e mettendo a nero tutti i pixel il cui valore è minore a T e a bianco tutti gli altri.



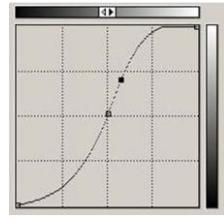
Variazioni di contrasto

Aumentare il contrasto significa rendere più evidenti le differenze di colore. Ciò si ottiene andando a cambiare il valore di un pixel con un altro che sia più scuro o più chiaro.

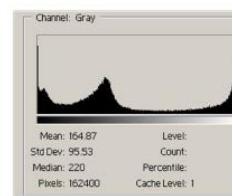
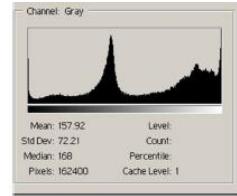
Aumento del contrasto



Come cambia la curva



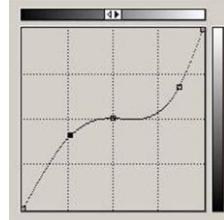
Come cambia l'istogramma



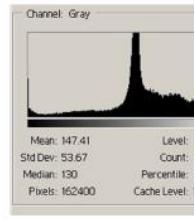
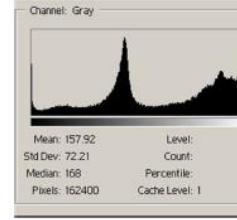
Diminuzione del contrasto



Come cambia la curva



Come cambia l'istogramma



Curve non monotone

È possibile fare delle variazioni alle curve in modo che questa diventi non monotona. Un esempio è la "solarizzazione"

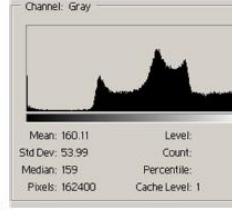
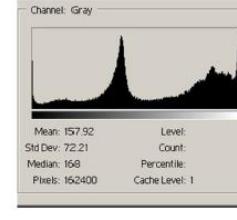
Diminuzione del contrasto



Come cambia la curva



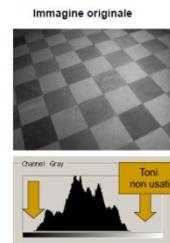
Come cambia l'istogramma



Contrast stretching (espansione del contrasto)

Serve per aumentare la dinamica di un'immagine il cui istogramma è concentrato su un intervallo limitato dei valori possibili.

Si ottiene spostando (con appositi algoritmi) i valori di un bin dell'istogramma verso un altro bin non utilizzato. L'istogramma apparirà in maniera differente, tipo pettine. Ciò è fatto per mettere in risalto che i bin mancanti sono stati distribuiti lungo altri livelli.



Aritmetica delle immagini

Operando aritmeticamente può accadere che un pixel abbia:

- Un valore negativo;
- Un valore maggiore del massimo (tipicamente 255);
- Un valore non intero (facilmente risolubile con una approssimazione o un troncamento);

Normalizzazione

I problemi a) e b) si chiamano **problemi di range**. Le soluzioni più comuni sono:

- Settare a 0 (colore nero) i valori negativi e a 255 (colore bianco) i valori maggiori di 255.
- Ri-normalizzare il range trasformando ciascun valore della matrice secondo l'equazione chiamata **Normalizzazione**:

$$v_{nuovo} = 255 * \frac{v_{vecchio} - min_{osservato}}{max_{osservato} - min_{osservato}}$$

Ovviamente, se solo un numero della matrice è fuori dal range [0..255], bisognerà normalizzare tutti i valori della matrice per non creare disparità e uniformare la normalizzazione.

Esempio. Normalizzare la matrice $\begin{bmatrix} -13 & 5 & 6 \\ 7 & 8 & 80 \end{bmatrix}$

$$v_{00} = 255 * \frac{-13+13}{80+13} = 0; \quad v_{01} = 255 * \frac{5+13}{80+13} = \frac{1530}{31} = 49,35 \dots = 49; \quad v_{02} = \dots$$

Equalizzazione

Si parla di immagine equalizzata quando il contributo di ogni differente tonalità di grigio è pressappoco eguale. Si parla anche di "istogramma" uniforme o appiattito. L'equalizzazione si ottiene usando appositi algoritmi. **Attenzione:** non sempre l'equalizzazione migliora l'immagine!

Il numero di livelli di grigio di un'immagine è dato da $2^{\#bit}$. Se r_k è un livello di grigio e n_k il numero di pixel nell'immagine $M \times N$ di quel livello di grigio, si può definire

$$p_r(r_k) = \frac{n_k}{MN} \quad k = 0, 1, 2, \dots, L-1$$

Se facciamo il plot di r_k versus $p_r(r_k)$ quello che si ottiene è l'istogramma dell'immagine.

I nuovi valori di grigio dell'istogramma sono così definiti:

$$s_k = T(r_k) = (L-1) \sum_{j=0}^k p_r(r_j) = \frac{L-1}{MN} \sum_{j=0}^k n_j \quad k = 0, 1, 2, \dots, L-1$$

Esempio. Sia data un'immagine a 3 bit ($L=8$) con 64×64 pixel ($MN=4096$) con la seguente distribuzione di intensità:

Applicando la formula si ha:

$$s_0 = T(r_0) = 7 \sum_{j=0}^0 p_r(r_j) = 7p_r(r_0) = 1.33$$

$$s_1 = T(r_1) = 7 \sum_{j=0}^1 p_r(r_j) = 7p_r(r_0) + 7p_r(r_1) = 3.08$$

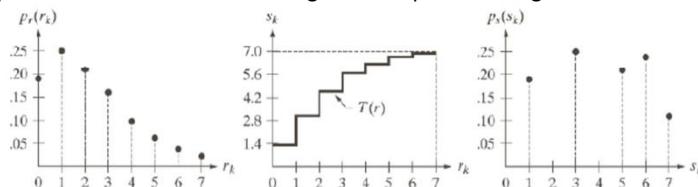
$$s_2 = 4.55, s_3 = 5.67, s_4 = 6.23, s_5 = 6.65, s_6 = 6.86, s_7 = 7.00.$$



r_k	n_k	$p_r(r_k) = n_k/MN$
$r_0 = 0$	790	0.19
$r_1 = 1$	1023	0.25
$r_2 = 2$	850	0.21
$r_3 = 3$	656	0.16
$r_4 = 4$	329	0.08
$r_5 = 5$	245	0.06
$r_6 = 6$	122	0.03
$r_7 = 7$	81	0.02

$$\text{arrotondando:} \quad \begin{aligned} s_0 &= 1.33 \rightarrow 1 & s_4 &= 6.23 \rightarrow 6 \\ s_1 &= 3.08 \rightarrow 3 & s_5 &= 6.65 \rightarrow 7 \\ s_2 &= 4.55 \rightarrow 5 & s_6 &= 6.86 \rightarrow 7 \\ s_3 &= 5.67 \rightarrow 6 & s_7 &= 7.00 \rightarrow 7 \end{aligned}$$

Questi sono i valori dell'istogramma equalizzato. Dato che $r_0 = 0$ è stato trasformato in $s_0 = 1$, ci sono 790 pixel nell'immagine dell'istogramma equalizzato con questo valore. Inoltre ci sono 1023 pixel con valore di $s_1 = 3$, 850 pixel con valore $s_2 = 5$, ecc. Dividendo questi numeri per $MN = 4096$ si ottiene l'istogramma equalizzato seguente:



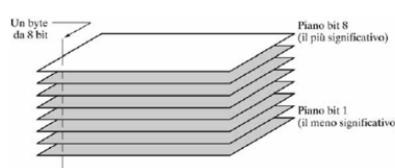
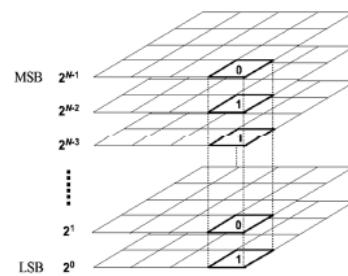
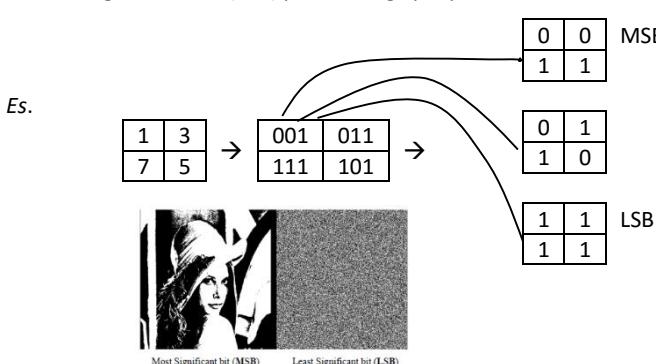
Esempio. Equalizzare la seguente matrice di un immagine $\begin{bmatrix} 2 & 3 & 4 & 2 & 3 \\ 5 & 2 & 1 & 6 & 7 \\ 7 & 3 & 4 & 3 & 1 \\ 5 & 0 & 5 & 4 & 4 \\ 2 & 6 & 4 & 1 & 2 \end{bmatrix}$ data la formula $s_k = \frac{L-1}{MN} \sum_{j=0}^k n_j$

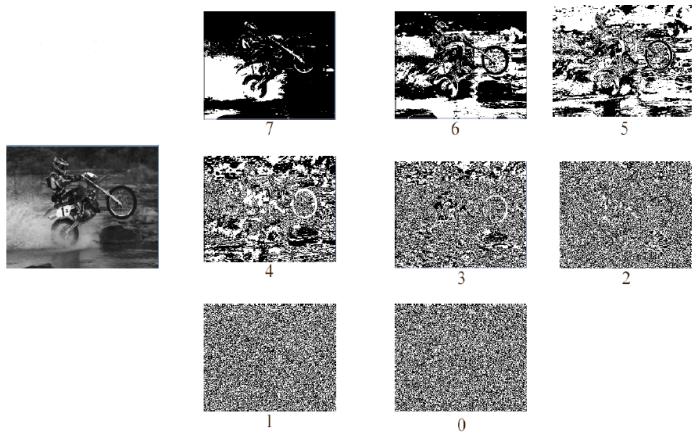
Al livello 0 c'è un pixel, ovvero nella matrice solo un elemento ha valore 0, ecc.

$$\begin{aligned} s_0 &= 1 & s_0 &= \frac{7}{25} \cdot 1 = 0 \\ s_1 &= 3 & s_1 &= \frac{7}{25} \cdot (1+3) = 1 \\ s_2 &= 4 & s_2 &= \frac{7}{25} \cdot (1+3+4) = 2 \\ s_3 &= 4 & s_3 &= \frac{7}{25} \cdot (1+3+4+4) = 3 \\ s_4 &= 5 & s_4 &= \dots \\ s_5 &= 3 & s_5 &= \dots \\ s_6 &= 2 & s_6 &= \dots \\ s_7 &= 2 & s_7 &= \dots \end{aligned} \quad (\text{i valori risultati sono arrotondati})$$

Bit-plane

Un'immagine con una profondità colore di N bit può essere rappresentata da N piani di bit (bit-planes), ciascuno dei quali può essere vista come una singola immagine binaria. In particolare si può indurre un ordine che varia dal **Most Significant Bit (MSB)** per i dettagli maggiori, fino al **Least Significant Bit (LSB)** per i dettagli più piccoli.





I piani di bit più significativi contengono informazioni sulla struttura dell'immagine, mentre quelli via via meno significativi forniscono i dettagli sempre più piccoli. Si noti che solo i piani dal 7 al 3 contengono dati significativi dal punto di vista visuale.

Il rumore delle immagini e gli errori di acquisizione sono più evidenti nei piani bassi.

Questo genere di scomposizione è molto utile per eliminare tutti i valori compresi in un certo range.

Ad esempio, se si vogliono eliminare tutti i grigi compresi tra 32 e 64, è necessario porre a 0 il quinto bit, e quindi tutto il piano 5.

<i>Ese.</i>	<table border="1"> <tr><td>6</td><td>3</td></tr> <tr><td>2</td><td>4</td></tr> </table>	6	3	2	4	\rightarrow	<table border="1"> <tr><td>6=110</td><td>3=011</td><td>2=010</td><td>4=100</td></tr> </table>	6=110	3=011	2=010	4=100	\rightarrow	<table border="1"> <tr><td>1</td><td>0</td></tr> <tr><td>0</td><td>1</td></tr> </table>	1	0	0	1	<table border="1"> <tr><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td></tr> </table>	1	1	1	0	<table border="1"> <tr><td>0</td><td>1</td></tr> <tr><td>0</td><td>0</td></tr> </table>	0	1	0	0	MSB	LSB
6	3																												
2	4																												
6=110	3=011	2=010	4=100																										
1	0																												
0	1																												
1	1																												
1	0																												
0	1																												
0	0																												

Convoluzioni e loro proprietà

Base canonica

In generale, una base di uno spazio vettoriale è un insieme di vettori linearmente indipendenti che generano lo spazio. Una base canonica è caratterizzata da avere tutti elementi uguali a zero, tranne uno che è uguale a 1. Dato un vettore di lunghezza N, questo può essere pensato come un elemento di uno spazio N dimensionale.

Es. il vettore [100,250,10] può essere scomposto in $[100,250,10] = 100*[1,0,0] + 250*[0,1,0] + 10*[0,0,1]$.

Lo stesso ragionamento vale per le immagini:

$$\begin{array}{|c|c|c|} \hline 1 & 2 & 3 \\ \hline 4 & 5 & 6 \\ \hline 7 & 8 & 9 \\ \hline \end{array} = 1 * \begin{array}{|c|c|c|} \hline 1 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} + 2 * \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} + 3 * \begin{array}{|c|c|c|} \hline 0 & 0 & 1 \\ \hline 0 & 0 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array} + \dots$$

Operatori locali

Il valore d'uscita di ogni pixel dipende da un limitato intorno del corrispondente punto in input. Sono usati per migliorare la qualità delle immagini o per estrarre delle informazioni dall'immagine. Si possono pensare come **filtraggi** dell'immagine. Un filtraggio è ottenuto facendo la convoluzione tra l'immagine ed una matrice.

Operatori lineari

Un operatore $F: V \rightarrow W$ si dice lineare se per ogni coppia di vettori v_1 e v_2 in V e per ogni coppia di scalari a, b si ha che $F(a v_1 + b v_2) = a F(v_1) + b F(v_2)$

Conseguenza: se conosco una base di V ed il comportamento dell'operatore F su ogni elemento di tale base, posso calcolare il comportamento di F su ogni elemento di V .

Es. Sia F un operatore da R^3 a R^3 che trasforma ogni vettore (x,y,z) in $(x/2, y/3, z/4)$. Esso è lineare poiché è completamente descritto una volta che si descrive il suo comportamento su ogni elemento della base canonica:

$$\begin{aligned} F((1,0,0)) &= (1/2, 0, 0); \\ F((0,1,0)) &= (0, 1/3, 0); \\ F((0,0,1)) &= (0, 0, 1/4); \end{aligned}$$

L'operatore puntuale "negativo" non è lineare poiché

$$F((x,y,z)) = (255-x, 255-y, 255-z)$$

È diverso da

$$\begin{aligned} &x * F((1,0,0)) + y * F((0,1,0)) + z * F((0,0,1)) \\ &= x * (255-1, 0, 0) + y * (0, 255-1, 0) + z * (0, 0, 255-1) \\ &= x * (255-1) + y * (255-1) + z * (255-1) \end{aligned}$$

Questo esempio ha un comportamento che non è lo stesso su tutti gli elementi della base canonica di R^N . Infatti il comportamento varia da elemento ad elemento a seconda della posizione all'interno della immagine. Questo è un operatore NON invariante per traslazioni!

Dunque per descrivere questi operatori, dobbiamo conoscere il suo comportamento su ciascun "impulso" in ciascuna locazione delle immagini.

Es. $F(x) = 3 \rightarrow F(av_1 + bv_2) = aF(v_1) + bF(v_2) \rightarrow 3(av_1 + bv_2) = a3v_1 + b3v_2 \rightarrow 3(av_1 + bv_2) = 3(av_1 + bv_2)$ è lineare

Operatori invarianti per traslazione

Un operatore si dice invariante per traslazione (shift invariant) quando il suo comportamento sulle immagini impulsive è sempre il medesimo indipendentemente dalla posizione in cui si trova il pixel.

Tutti gli operatori puntuali sono invarianti per traslazione (anche se non sono lineari).

Schema di descrizione di F

- **F lineare:** per descriverlo basta conoscere il comportamento su tutte le immagini impulsive
- **F shift invariant:** si comporta allo stesso modo su tutti gli impulsi, indipendentemente dalla loro posizione
- **F lineare e shift invariant:** per descriverlo basta conoscere come si comporta su un solo impulso.

La "risposta all'impulso" o "point spread function" di F è la carta di identità di tale operatore.

Ad un operatore lineare e shift invariante corrisponde una **maschera** ma vale anche il viceversa: ad una maschera corrisponde un simile operatore.

Es. Si consideri l'operazione che preso un impulso:

$$\begin{array}{|c|} \hline \blacksquare \\ \hline \end{array} \quad \begin{array}{c} 0 \ 0 \ 0 \ 0 \\ 0 \ 0 \ 1 \ 0 \\ 0 \ 0 \ 0 \ 0 \\ \hline \end{array} \quad \text{lo trasforma in:} \quad \begin{array}{c} 0 \ 0 \ 0 \ 0 \ 0 \\ 0 \ .5 \ 0 \ .5 \ 0 \\ 0 \ 0 \ 0 \ 0 \ 0 \\ \hline \end{array} \quad \begin{array}{|c|c|} \hline \blacksquare & \blacksquare \\ \hline \end{array}$$

Tale "risposta all'impulso" o PSF definisce completamente un operatore lineare e invariante per traslazioni F . Spesso un operatore su una immagine prende il nome di "**filtro**".

La matrice che descrive la risposta all'impulso si chiama anche **kernel** o maschera dell'operatore.

Essa è detta anche maschera di convoluzione di F per ragioni che vedremo tra breve.

Kernel finiti o infiniti

La grandezza del kernel può variare fino ad essere infinita. Per ragioni pratiche, però, si usano solo kernel con dimensioni finite. Le dimensioni del kernel influenzano la complessità della operazione di filtraggio. Tale complessità dipende ovviamente anche dal numero dei pixel di una immagine.

Filtri convolutivi

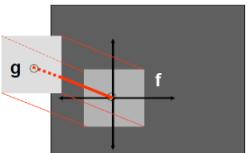
I filtri lineari e invarianti per traslazione vengono chiamati anche filtri convolutivi. Dobbiamo studiare la operazione di convoluzione per capire meglio come un filtro può essere calcolato. Inoltre la convoluzione è un fenomeno estremamente importante per ogni tipo di signal processing e per la descrizione di numerosi eventi fisici.

Convoluzione

In matematica, in particolare nell'analisi funzionale, la convoluzione è un'operazione tra due funzioni che genera una terza funzione che viene vista come la versione modificata di una delle due funzioni di partenza

Proprietà

Per indicare l'operazione di convoluzione si usa la notazione $h = f \otimes g$
essa è commutativa $f \otimes g = g \otimes f$ ed associativa $(f \otimes g) \otimes h = f \otimes (g \otimes h)$



Convoluzione nel caso finito

Se il kernel f ha dimensioni $k \times h$ e se gli indici del kernel sono disposti in modo da avere il punto di coordinate (0,0) nella posizione centrale, la formula va riscritta nella seguente maniera

$$h_{m,n} = \sum_{i=-\left[\frac{k}{2}\right]}^{\left[\frac{k}{2}\right]-1} \sum_{j=-\left[\frac{h}{2}\right]}^{\left[\frac{h}{2}\right]-1} (f_{i,j} * g_{m+i,n+j})$$

-1	0	1
a	b	c
d	e	f

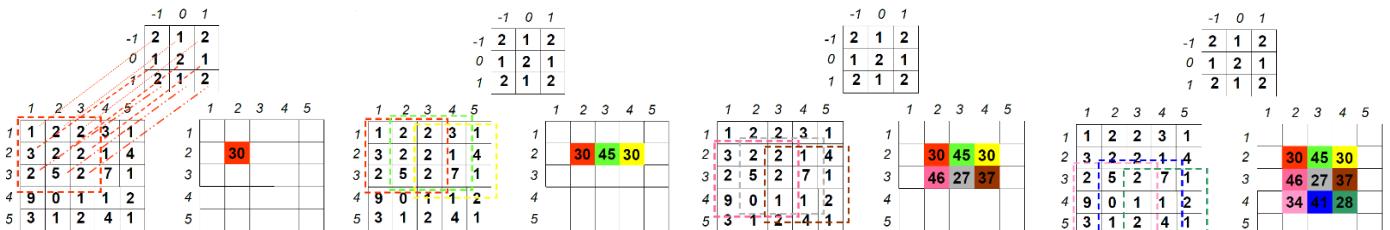
1	2	3
a	b	c
d	e	f

1	2	3
g	h	i

Se il kernel f ha dimensioni $k \times h$ e se gli indici del kernel sono disposti partendo da 1 fino ad arrivare ad h o k , la formula va riscritta nella seguente maniera

$$h_{m,n} = \sum_{i=1,j=1}^{k,h} f_{i,j} * g_{m+(i-k+\left[\frac{k}{2}\right]),n+(j-h+\left[\frac{h}{2}\right])}$$

Esempio:



Convoluzione e filtraggio

Applicare un filtro lineare e shift invariante ad una immagine è equivalente a calcolare la convoluzione del kernel del filtro con l'immagine.

Problema dei bordi

Un problema è quello dei bordi: come fare la convoluzione e il filtraggio ai bordi? Analizziamo le possibili soluzioni:

Filtrare solo le zone centrali dell'immagine

input					output				
<small>Le aree in grigio non verranno calcolate</small>									
1	2	2	3	1					
3	2	2	1	4					
2	5	2	7	1					
9	0	1	1	2					
3	1	2	4	1					

Assumere una topologia "toroidale": quando si "sfiora a destra" si rientra a sinistra, quando si "sfiora" in basso di rientra in alto e viceversa;

input					output				
1	3	1	2	4	1	3			
1	1	2	2	3	1	1			
4	3	2	2	1	4	3			
1	2	5	2	7	1	2			
2	9	0	1	1	2	9			
1	3	1	2	4	1	3			
1	1	2	2	3	1	1			

Supporre che tutto intorno all'immagine ci sia 0

input										output									
0	0	0	0	0	0	0	0	0	0	11	19	17	22	11	25	30	45	30	31
0	1	2	2	3	1	0				25	46	27	37	19	35	34	41	28	29
0	3	2	2	1	4	0				16	27	12	18	10					
0	2	5	2	7	1	0													
0	9	0	1	1	2	0													
0	3	1	2	4	1	0													
0	0	0	0	0	0	0													

Aggiungere una riga all'inizio uguale alle riga precedente, una riga alla fine uguale all'ultima riga, una colonna all'inizio uguale alla colonna iniziale, e una colonna alla fine uguale alla colonna finale.

input										output									
1	1	2	2	3	1	1				25	27	29	31	33	34	30	45	30	39
3	3	2	2	1	4	4				51	46	27	37	32	54	34	41	28	35
2	2	5	2	7	1	1				48	32	24	34	26					
9	9	0	1	1	2	2													
3	3	1	2	4	1	1													
3	3	1	2	4	1	1													

Esempio:

if (filtri lineari && invarianti per traslazione)

 filtri convolutivi

else

 filtri non convolutivi

$$3 \cdot 0 + 5 \cdot 0 + 6 \cdot 1 + 3 \cdot 1 + 1 \cdot 0 + \dots = x$$

kernel

$$\begin{bmatrix} 3 & 5 & 6 & 3 \\ 3 & 1 & 1 & 2 \\ 2 & 3 & 4 & 0 \end{bmatrix}$$

filtro da applicare

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 3 \\ 5 & 4 & 2 \end{bmatrix}$$

prendo il primo intorno

$$\begin{bmatrix} 3 & 5 & 6 & 3 \\ 3 & 1 & 1 & 2 \\ 2 & 3 & 4 & 0 \end{bmatrix}$$

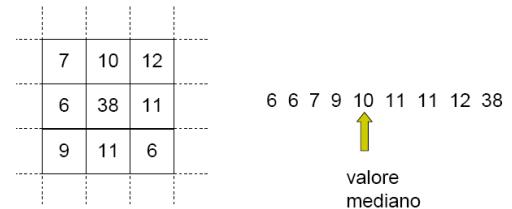
$$\text{Kernel} \quad \begin{bmatrix} ? & ? & ? & ? \\ ? & x & ? & ? \\ ? & ? & ? & ? \end{bmatrix}$$

I bordi gli amplio con 0 e gli applico la formula della convoluzione per trovarli

Esempi di operatori locali

Mediano

È un filtro non lineare che fornisce in uscita il valore mediano dell'intorno del pixel.



Filtro di minimo e filtro di massimo

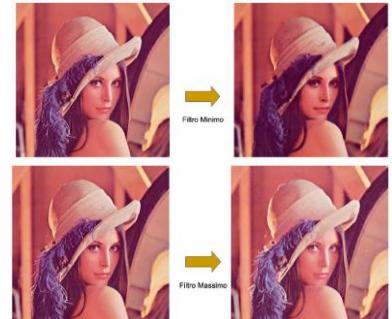
Oltre al filtro mediano esistono altri filtri statistici detti "order statistics".

Il filtro di minimo preso un intorno $m \times m$ di un pixel (con m generalmente dispari), sostituisce il valore del pixel con il valore minimo di tutti i valori osservati in tale intorno.

Il filtro di massimo preso un intorno $m \times m$ di un pixel (con m generalmente dispari), sostituisce il valore del pixel con il valore massimo di tutti i valori osservati in tale intorno.

Se si sostituisce con il minimo si ottiene un incupimento dell'immagine (si eliminano per esempio macchie chiare);

Se si sostituisce con il massimo si ottiene uno schiarimento dell'immagine (si eliminano per esempio punti neri).



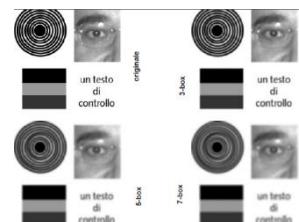
Filtro N-box (o di media)

Sono definiti da kernel $N \times N$ con ogni elemento pari a $1/N^2$. Si sceglie generalmente un valore N dispari.

Hanno l'effetto di sfocare le immagini. La sfocatura è molto forte in orizzontale e verticale ma meno in diagonale.

Es.

$$\begin{array}{l} \text{3-box} \\ \text{1/9} * \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \end{array} \rightarrow \begin{array}{l} \text{5-box} \\ \text{1/25} * \begin{array}{|c|c|c|c|c|} \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline \end{array} \end{array}$$



Filtro N-binomiale

Sono filtri di smussamento con kernel derivati dalla distribuzione binomiale. Poiché tale distribuzione è una approssimazione discreta della distribuzione gaussiana sono anche detti filtri gaussiani. Hanno il pregio di smussare egualmente in tutte le direzioni. Smussano meno vigorosamente degli n-box.

3-binomiale					5-binomiale				
$1/16 *$					$1/256 *$				
1	2	1			1	4	6	4	1
2	4	2			4	16	24	16	4
1	2	1			6	24	36	24	6

Conservazione dell'energia nei filtri

Con abuso di linguaggio derivato dalla Fisica, si dice che un filtro "conserva l'energia" se la somma dei suoi pesi fa 1.

Tutti i filtri di smoothing visti prima sono "energy preserving": la somma dei valori totali della luminanza nella immagine non cambia.

Esistono anche filtri "non energy preserving"

Noise cleaning e smoothing

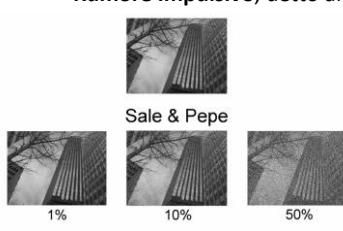
I filtri appena visti servono anche a ridurre il rumore in una immagine. In questo caso, più è grande il kernel e migliore sarà il risultato.

I filtri N-box e N-binomiali sono anche usati per sfocare l'immagine (**smoothing**). In questo caso, più è grande il kernel e maggiore sarà la sfocatura.

Rumore

Ci sono due tipi principali di rumore:

- **Rumore impulsivo**, detto anche "salt e pepe". Viene caratterizzato dalla frazione dell'immagine modificata (in %);



Rimozione del rumore



Sale & Pepe



Filtro Mediano 5x5



Filtro 5-box

- **Rumore gaussiano bianco.** Viene caratterizzato dalla media e dalla varianza.

Gaussiano



M=0,3 Var=0,02



M=0,1 Var=0,2



M=0,1 Var=0,02

Rimozione del rumore



Gaussiano



Filtro Mediano 5x5



Filtro 5-box

Media vs Mediano

Perché i filtri mediani danno risultati migliori rispetto a quelli di media?

- Il filtro media tende a creare dei livelli di grigio prima non esistenti.
- Il filtro di media non attenua solo il rumore ma anche tutte le alte frequenze spaziali in maniera indiscriminata dando origine ad immagini sfocate.
- Il filtro mediano non deteriora i lati, ma elimina i picchi con base piccola rispetto al kernel.

Altri filtri

Esistono altri filtri non lineari molto importanti:

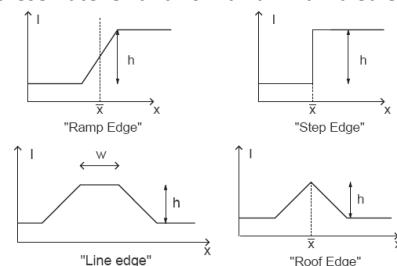
Outlier: il valore del pixel centrale viene confrontato con il valore della media dei suoi 8 vicini. Se il valore assoluto della differenza è maggiore di una certa soglia, allora il punto viene sostituito dal valore medio, altrimenti non viene modificato.

Olimpico: da un dato intorno si scartano i valori massimo e minimo e sul resto si fa la media.

Estrazione dei contorni

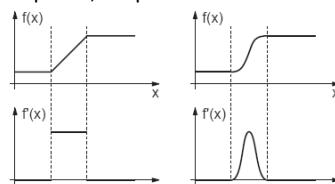
Gli operatori locali ci aiutano ad estrarre i contorni da una immagine. I contorni sono definiti come delle discontinuità locali della luminanza.

Gli *edge detector* forniscono immagini in cui sono preservate le variazioni di luminanza ed eliminate tutte le altre informazioni.



Edge detector basati sulla derivata prima

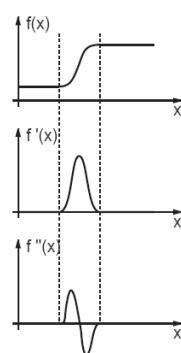
Se ho un segnale monodimensionale e calcolo la derivata prima, scopro che i lati sono i corrispondenti dei massimi della derivata.



Quindi i filtri devono calcolare la derivata in direzione x quella in direzione y e poi combinarle insieme.

Edge detector basati sulla derivata seconda

Se ho un segnale monodimensionale e calcolo la derivata seconda, scopro che in corrispondenza del lato essa passa per lo zero.



Kernel notevoli

- Lati orizzontali

$$Sobel_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad Prewitt_x = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$



- Lati verticali

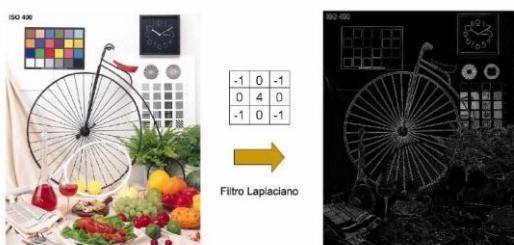
$$Sobel_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{Prewitt}_y = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$



Migliori risultati si ottengono con algoritmi più sofisticati (non lineari) per il calcolo della grandezza del gradiente (somma del quadrato della risposta di un edge finder orizzontale e del quadrato della risposta di un edge finder verticale)
Si ottengono con strategie più “intelligenti” (algoritmo di Canny, algoritmi fuzzy, tecniche di backtracking eccetera)

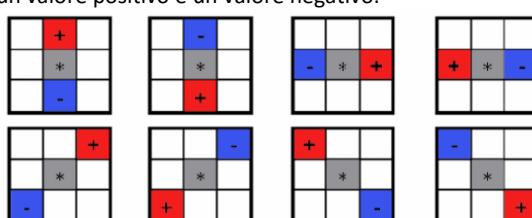
- Laplaciano

Il filtro più diffuso per calcolare la derivata seconda è detto Laplaciano, ed è definito dalla maschera:



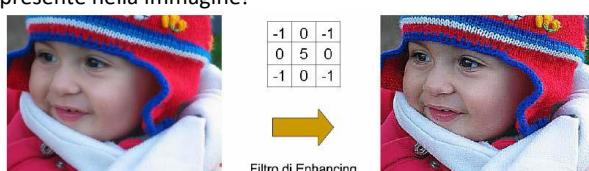
$$\text{Laplaciano} = \begin{bmatrix} -1 & 0 & -1 \\ 0 & 4 & 0 \\ -1 & 0 & -1 \end{bmatrix}$$

Dopo aver applicato l’operatore Laplaciano è necessario che si verifichi la condizione di **Zero-crossing**. Cioè, deve sempre accadere che rispetto al punto in questione ci sia nel suo intorno un valore positivo e un valore negativo.



Filtri di sharpening

Sono filtri il cui scopo è quello di incrementare la nitidezza di una immagine aumentando il contrasto locale. Questa è una operazione opposta allo sfocamento. Per ottenere tale effetto si può adottare una maschera che, derivata dal Laplaciano, “rinforza” i lati presenti nell’immagine. Purtroppo essa rinforza anche il rumore presente nella immagine!



Unsharp mask: un algoritmo non lineare

L’uso del kernel derivato dal Laplaciano per fare edge enhancing ha un difetto: esso viene applicato SEMPRE a tutti i pixel dell’immagine. Come conseguenza esso accentua i “dettagli” anche nelle zone omogene mettendo in evidenza i difetti ed il rumore. Sono stati proposti vari algoritmi per evitare questo fenomeno. La famiglia di algoritmi chiamata unsharp mask è la più famosa usata per questo scopo.

Serie e Trasformata di Fourier

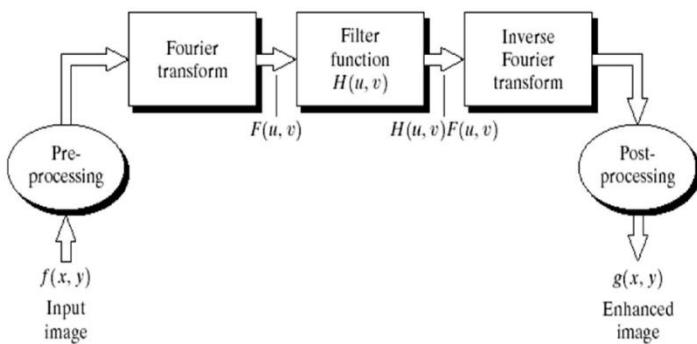
La serie di Fourier è una rappresentazione di una funzione periodica mediante una combinazione lineare di funzioni sinusoidali fondamentali.

La **trasformata di Fourier** è largamente utilizzata nell'analisi in frequenza dei sistemi dinamici, nella risoluzione delle equazioni differenziali e in teoria dei segnali. Il motivo di una così vasta diffusione risiede nel fatto che si tratta di uno strumento che permette di scomporre e successivamente ricombinare, tramite la formula inversa di sintesi o **antitrasformazione**, un segnale generico in una somma infinita di sinusoidi con frequenze, ampiezze e fasi diverse. L'insieme di valori in funzione della frequenza, continuo o discreto, è detto spettro di ampiezza e spettro di fase.

- Se il segnale in oggetto è un segnale **periodico**, la sua trasformata di Fourier è un insieme discreto di valori, che in tal caso prende il nome di spettro discreto o spettro a pettine. In questo caso la rispettiva formula inversa di sintesi costituisce lo sviluppo in serie di Fourier della funzione o segnale periodico originario.
- Nel caso in cui la funzione sia **non periodica** lo spettro è continuo

La teoria della trasformata e antitrasformata di Fourier generalizza dunque la teoria della Serie di Fourier al caso di segnali non periodici, ricomprensivo i segnali periodici come caso particolare ed insieme confluiscono nella cosiddetta Analisi di Fourier o analisi armonica.

Image Enhancement nel Dominio delle Frequenze



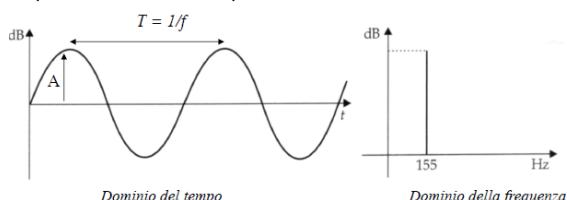
Una funzione periodica può essere espressa come somma di seni e/o coseni di differenti frequenze e ampiezze (**Serie di Fourier**). Anche una funzione non periodica, (sotto certe condizioni) può essere espressa come integrale di seni e/o coseni, moltiplicati per opportune funzioni-peso (**Trasformata di Fourier**).

Sia la serie di Fourier che la Trasformata di Fourier condividono il fatto che una funzione possa essere "ricostruita" (*recovered*) con un semplice processo di inversione senza perdita di informazione. E' cioè possibile lavorare nel cosiddetto **dominio di Fourier** e tornare nel dominio originale della funzione in maniera del tutto naturale.

Con l'avvento negli anni 60 della **FFT (Fast Fourier Transform)** il settore dell'**elaborazione digitale dei segnali (DSP – Digital Signal Processing)** ha subito una vera e propria rivoluzione, ed oggi questi concetti trovano applicazione nei più svariati campi industriali, dalla medicina, alle telecomunicazioni, ecc. dato che questo processo ritorna un'immagine (segnale) rinforzata di molto rispetto all'originale.

Immagini e segnali

Un'immagine può essere vista come una funzione discreta in due dimensioni i cui valori rappresentano il livello di grigio di un determinato pixel. La funzione "immagine" può essere vista come un segnale, cioè una funzione variabile in un dominio con una propria frequenza (costante o variabile).



- Ampiezza (A) espressa in decibel dB;
- Periodo (T) espresso in secondi;
- Frequenza (f) numero di cicli (onde) al secondo; si misura in Hertz Hz

Discrete Fourier Transform

La relazione tra Δu e Δx è la seguente: $\Delta u = \frac{1}{N\Delta x}$

Nel caso 2-D la coppia trasformata antitrasformata della sequenza bidimensionale $f(x,y)$ assume la seguente forma:

$$F(u, v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-i2\pi(\frac{ux}{M} + \frac{vy}{N})} \quad \text{per } u = 0, 1, \dots, M-1 \quad v = 0, 1, \dots, N-1$$

$$f(x, y) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} F(u, v) e^{i2\pi(\frac{ux}{M} + \frac{vy}{N})} \quad \text{per } u = 0, 1, \dots, M-1 \quad v = 0, 1, \dots, N-1$$

u e v sono gli indici relativi agli assi frequenze discretizzati, mentre M e N sono le dimensioni (in pixel) dell'immagine. Il campionamento della $f(x,y)$ ha luogo nei punti di una griglia bidimensionale, con passi Δx e Δy . Per la $F(u,v)$ valgono considerazioni analoghe a quelle fatte nel caso monodimensionale.

Trasformata di Fourier

La trasformata di Fourier serve per passare dal piano spaziale a quello delle frequenze. Dato che la trasformata F ha valori complessi, può essere espressa in termini della sua parte reale e della sua parte immaginaria.

Spettro della Trasformata

$$|F(u, v)| = \sqrt{R^2(u, v) + I^2(u, v)}$$

Angolo di Fase

$$\phi(u, v) = \tan^{-1} \left[\frac{I(u, v)}{R(u, v)} \right]$$

Potenza Spettrale

$$P(u, v) = |F(u, v)|^2 = R^2(u, v) + I^2(u, v)$$

Esempio 1-D

$$F(0) = \sum_{x=0}^3 f(x) = [f(0) + f(1) + f(2) + f(3)] = 1 + 2 + 4 + 4 = 11$$

$$F(1) = \sum_{x=0}^3 f(x)e^{-\frac{j2\pi(1)x}{4}} = 1e^0 + 2e^{-\frac{j\pi}{2}} + 4e^{-j\pi} + 4e^{-\frac{j3\pi}{2}} = -3 + 2j$$

$$F(2) = -(1 + 0j) \quad ed \quad F(3) = -(3 + 2j)$$

Se invece prendiamo $F(u)$ e vogliamo calcolarne la sua inversa, dobbiamo procedere alla stessa maniera, stavolta utilizzando l'antitrasformata

$$f(0) = \frac{1}{4} \sum_{u=0}^3 F(u)e^{j2\pi u(0)} = \frac{1}{4} \sum_{u=0}^3 F(u) = \frac{1}{4}[11 - 3 + 2j - 1 - 3 - 2j] = \frac{1}{4}[4] = 1$$

Tutti i valori della $f(x)$ contribuiscono alla costruzione di ciascuno dei campioni della $F(u)$. Analogamente, tutti i campioni della trasformata contribuiscono, durante la antitrasformata, a ciascuno dei valori della $f(x)$. I campioni della $F(u)$ sono in genere complessi, per cui ciascuno di essi ha un modulo e una fase.

Range dinamico

Quando si visualizza lo spettro di Fourier come immagine di intensità, esso manifesta in genere una dinamica molto più grande di quella riproducibile su un tipico display, per cui solo le parti più luminose dello spettro risultano visibili.

Per esempio, lo spettro dell'immagine di Lena varia tra 0 (circa) e 6.47×10^6 . Effettuando la normalizzazione necessaria per visualizzarlo con $L=256$ livelli di grigio, solo pochissime parti molto luminose sono visibili.

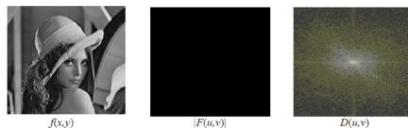
A ciò si può ovviare, come è noto, mediante una compressione di tipo logaritmico, visualizzando, invece che lo spettro, una funzione del tipo:

$$D(u, v) = c \log(1 + |F(u, v)|)$$

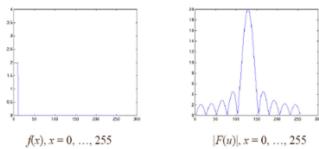
c è una costante di scala, che va scelta opportunamente per far ricadere i valori trasformati nel range voluto, cioè in $[0, L-1]$

Poiché $0 < |F(u, v)| < R = 6.47 \times 10^6$, si ha $0 < D(u, v) < c \log(1+R)$. Dato che $R >> 1$, come peraltro avviene normalmente per lo spettro di Fourier di una immagine, si può porre $c \log R = L-1$, da cui $c = (L-1)/\log R = 255/\log(6.47 \times 10^6) = 16.26$

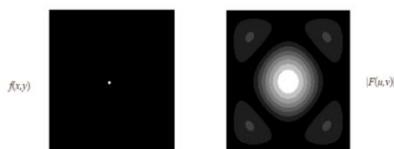
Pertanto $D(u, v)$ ha tutti i valori nell'intervallo $[0, 255]$, e ciò consente la visualizzazione di molti più dettagli.



Es. Un esempio di trasformata discreta nel caso 1-D: un impulso approssimato da un rettangolo di lato 10 e altezza 2, su una finestra complessiva di 256 valori di x :



Es. Un esempio di trasformata discreta nel caso 2-D: un impulso approssimato da un piccolo cerchio bianco su fondo nero, in un'immagine di circa 200 x 200 pixels. I differenti livelli di grigio nell'immagine di intensità dello spettro evidenziano le ampiezze decrescenti dei diversi lobi.



Trasformate di Fourier: vantaggi

Nello spazio delle frequenze è possibile:

- sopprimere frequenze indesiderate
- ridurre lo spazio occupato dai dati pur limitando la degenerazione del segnale (JPEG, MPEG, DivX, MP3)
- rigenerare segnali degradati

La trasformazione diretta può essere vista come un processo di **analisi**: il segnale $f(x)$ viene scomposto nelle sue componenti elementari, che sono nella forma dei vettori di base. I coefficienti della trasformata specificano quanto di ogni componente di base è presente nel segnale.

Nella trasformazione inversa, mediante un processo di sintesi, il segnale viene ricostruito, come somma pesata delle componenti di base: il peso di ogni vettore di base nella ricostruzione del segnale è rappresentato dal corrispondente coefficiente della trasformata.

Il coefficiente della trasformata è una misura della correlazione tra il segnale ed il corrispondente vettore di base. La trasformazione non comporta perdita di informazione: essa fornisce solo una rappresentazione alternativa del segnale originale.

Proprietà della DFT 2D

Separabilità

La trasformata di Fourier discreta può essere espressa in forma separabile. In particolare vale la seguente espressione:

$$F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} g(x, v) e^{-\frac{\pi u x}{N}} \quad \text{dove} \quad g(x, v) = N \left[\frac{1}{N} \sum_{y=0}^{N-1} f(x, y) e^{-\frac{\pi v y}{N}} \right]$$

Il principale vantaggio delle proprietà di separabilità è che la $F(u, v)$ può essere ottenuta applicando in due passi successivi la trasformata 1D.

Traslazione

È possibile dimostrare che

$$f(x, y) e^{\frac{i2\pi(u_0x+v_0y)}{N}} \Leftrightarrow F(u - u_0, v - v_0) \quad f(x - x_0, y - y_0) \Leftrightarrow F(u, v) e^{\frac{i2\pi(ux_0+vy_0)}{N}}$$

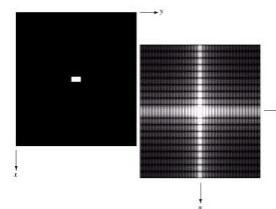
Cioè si trasla l'origine della DFT in (u_0, v_0) o si trasla l'origine della $f(x, y)$ in (x_0, y_0) .

Si dimostra inoltre che uno shift nella $f(x, y)$ non modifica la magnitudo della trasformata dato che:

$$\left| F(u, v) e^{\frac{-i2\pi(ux_0+vy_0)}{N}} \right| = |F(u, v)|$$

Queste proprietà vengono utilizzate per una migliore visualizzazione dello spettro.

La trasformata di un piccolo rettangolo bianco su uno sfondo nero sarà dunque come la figura a sinistra (dopo lo shift)



Valor Medio

Il valore della trasformata nell'origine, cioè nel punto $(u, v) = (0, 0)$ è dato da:

$$F(0, 0) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \quad \bar{f}(x, y) = \frac{1}{N} F(0, 0)$$

Come si può vedere non è altro che la media di $f(x, y)$. Il valore della trasformata di Fourier di un'immagine $f(x)$ nell'origine è uguale alla media dei valori di grigio contenuti nell'immagine. $F(0, 0)$ prende anche il nome di componente continua o componente DC.

Fast Fourier Transform

$$F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x) \exp \left[-\frac{i2\pi u}{N} \right]$$

Nella sua forma classica implementare la trasformata di Fourier richiederebbe un numero di operazioni proporzionale a N^2 (N moltiplicazioni complesse e $N-1$ addizioni per ciascuno degli N valori di u). Utilizzando opportune tecniche di decomposizione è possibile abbassare la complessità a $N \log_2 N$, implementando la cosiddetta Fast Fourier Trasform (FFT).

Teorema della Convoluzione

La convoluzione di due segnali nel dominio spaziale equivale all'antitrasformata del prodotto delle frequenze.

Il fondamento teorico delle tecniche di elaborazione nel dominio della frequenza, basate sulla manipolazione della DFT dell'immagine, è rappresentata dal teorema della convoluzione che fa corrispondere, alla operazione così definita nel dominio spaziale:

$$g(x, y) = f(x, y) * h(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) h(x - m, y - n)$$

L'operazione, nel dominio delle frequenze:

$$G(u, v) = F(u, v) H(u, v)$$

Complessità di un segnale 1D

- Nel dominio delle frequenze: $O(n \ log n)$
- Nel dominio spaziale: $O(n^2)$

Effettivamente vale la pena passare al dominio delle frequenze.

Filtraggio nel Dominio delle Frequenze

Se il filtro ha dimensioni confrontabili con quelle dell'immagine è più efficiente computazionalmente effettuare il filtraggio nel dominio delle frequenze. Con maschere più piccole diviene più efficiente il calcolo nel dominio spaziale. La definizione di un filtro nel dominio delle frequenze è più intuitiva.

Per ottenere un filtro a partire dalla maschera spaziale:

1. Il filtro H ha la stessa dimensione dell'immagine I ;
2. H deve avere in alto a sinistra i valori della maschera spaziale, nel resto sempre il valore 0;
3. Si fa lo shift di H
4. Si calcola da H la trasformata di fourier.

Filtri low pass nel dominio della frequenza

Low pass ideale

$$H(u, v) = \begin{cases} 1 & \text{se } D(u, v) \leq D_0 \\ 0 & \text{se } D(u, v) > D_0 \end{cases} \quad D(u, v) = \left[\left(u - \frac{P}{2} \right)^2 + \left(v - \frac{Q}{2} \right)^2 \right]^{\frac{1}{2}}$$

Filtro passa-basso di Butterworth

La funzione di trasferimento del filtro passa-basso di Butterworth di ordine n e frequenza di taglio D_0 è:

$$H(u, v) = \frac{1}{1 + \left[\frac{D(u, v)}{D_0} \right]^{2n}}$$

Filtro Gaussiano

$$H(u, v) = e^{-\frac{D^2(u, v)}{2D_0^2}}$$

I filtri gaussiani hanno il grande vantaggio di avere come trasformata di Fourier ancora una gaussiana.

Compressione

Algoritmo di Compressione

Un algoritmo di compressione è una tecnica che elimina la **ridondanza di informazione dai dati**, ovvero quegli elementi (come i dettagli) la cui eliminazione non comporta danno per la comprensione, e consente un risparmio di memoria.

Si possono comprimere TUTTI i tipi di dati che contengano al loro interno una certa ridondanza statistica: immagini, audio, video...

Tipi di compressione

Si distinguono due modalità di compressione:

- **Lossless (reversibile)**, cioè senza perdita di informazione:

Si parla di compressione LOSSLESS quando i dati possono essere trasformati in modo da essere memorizzati con risparmio di memoria e successivamente ricostruiti perfettamente, senza errore e senza perdita di alcun bit di informazione.

Tale tipo di compressione è ovviamente necessario per ridurre lo spazio occupato da documenti, programmi, eseguibili, eccetera

- **Lossy (irreversibile)**, con eventuale perdita di informazione:

Si parla di compressione LOSSY quando i dati possono essere trasformati in modo da essere memorizzati con risparmio di memoria ma con perdita di informazione. Tale tipo di compressione produce un maggiore risparmio di memoria.

Compressione Lossless

Run-Length-Encoding (RLE)

Si voglia comprimere la sequenza:

00000111001011101110101111111

Si potrebbe ricordare in alternativa:

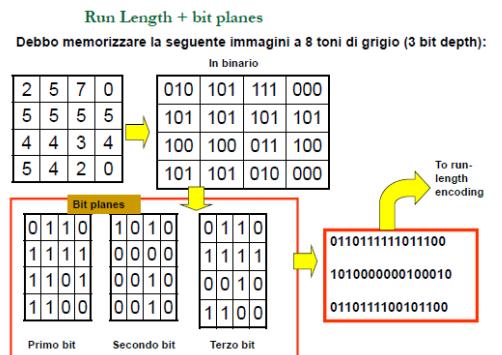
5 volte 0, 3 volte 1, 2 volte 0 etc.

O meglio basterebbe accordarsi sul fatto che si inizia con il simbolo 0 e ricordarsi solo la lunghezza dei segmenti (run) di simboli eguali che compongono la sequenza:

5,3,2,1,1,3,1,3,1,1,7

Tali valori vanno adesso scritti in binario. Non sempre tale codifica porta un risparmio rispetto a quella di input. Ciò accade solo se la lunghezza della run (sequenze di simboli eguali) è molto grande e prevede un numero di bit superiore a quelli necessari per scrivere il numero che rappresenta la run.

Se ci sono molte run piuttosto lunghe ricordare la sequenza delle loro lunghezze potrebbe portare un risparmio.



Teorema di Shannon per la compressione

Ma fino a che punto si può comprimere in maniera lossless? Non certo a piacere. C'è un teorema che ci indica in numero minimo di bit necessari per comprimere un segnale. Prima di introdurre il teorema occorre richiamare alcuni concetti.

- **Frequenza:** Sia data una sequenza S di N caratteri tratti da un alfabeto di M possibili caratteri: a_1, \dots, a_m . Sia f_i la frequenza del carattere a_i , cioè
$$f_i = \frac{\# \text{occorrenze } a_i}{N}$$
- **Entropia:** Definiamo entropia E della sequenza di dati S :
$$E = -\sum f_i \log_2(f_i) \quad i \in S$$
 I logaritmi daranno tutti un valore negativo, occorre quindi cambiare di segno il risultato della sommatoria

Teorema fondamentale della teoria dell'informazione: I dati possono essere rappresentati senza perdere informazione (lossless) usando almeno un numero di bit pari a $N * E$ dove N è il numero di caratteri mentre E è l'entropia.

Il teorema di Shannon fissa il numero minimo di bit, ma non ci dice come trovarli. Occorre usare un algoritmo che permetta di codificare i nostri caratteri usando esattamente il numero di bit ricavati con il teorema di Shannon. Un algoritmo che fa ciò è dovuto ad Huffman.

Codifica di Huffman

Huffman ha proposto un semplice algoritmo greedy che permette di ottenere un "dizionario" (cioè una tabella carattere-codifica_binaria) per una compressione quasi ottimale dei dati cioè pari al limite di Shannon con un eccesso di al più N bit.

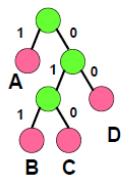
Es. Dati AABABCACAAADDD dove

A: Frequenza pari a $\frac{1}{2}$ B: Frequenza pari a $\frac{1}{8}$ C: Frequenza pari a $\frac{1}{8}$ D: Frequenza pari a $\frac{1}{4}$

L'algoritmo procede costruendo un albero binario le cui foglie sono i caratteri da codificare come segue:

Procedura: inizio con 4 alberi ciascuno composto di un singolo nodo. Aggrego i due alberi con minore frequenza e procedo in tal modo fino ad avere un solo albero che aggreghi tutte le foglie





Il risultato finale è un etichetta con 1 i rami sinistri e 0 i rami destri. Il cammino dalla radice al simbolo fornisce la parola del dizionario che "codifica" in maniera ottimale il simbolo.

Si osservi che ho parole di codice di varia lunghezza, ma nessuna è prefissa delle altre. Inoltre i simboli più frequenti richiedono meno bit, i meno frequenti più bit.

A:1 B:011 C:010 D:00

Dunque il codice per la sequenza AABABCAACAAADDD ha come codifica A:1; B:011; C:010; D:00
e la codifica della nostra stringa è: 1-1-011-1-011-010-1-1-010-1-1-00-00-00 pari a 28 bit

Si osservi che i trattini sono del tutto superflui perché nessun codice per i caratteri è prefisso degli altri, cioè posso decodificare senza fare errori se ho solo: 11011101101101011100000000

Quale è il limite previsto da Shannon?

$$16 * (-1/2 * \log_2(1/2) - 1/8 * \log_2(1/8) - 1/8 * \log_2(1/8) - 1/4 * \log_2(1/4)) = 16 * (1/2 + 3/8 + 3/8 + 2/4) = 8 + 6 + 6 + 8 = 28 \text{ bit} \rightarrow \text{CODIFICA OTTIMALE!}$$

Costo aggiuntivo: si deve memorizzare la tabella caratteri-codici. Se i caratteri sono tanti questo può essere costoso. Per le immagini a toni di grigio i "caratteri" sono i livelli di grigio (256) e tale tabella è assai poco pratica.

Huffman viene usato per comprimere alcune informazioni nella fase finale della codifica JPEG (dopo che è stata fatta una riduzione con altre tecniche).

Compressione Lossy

Gli uomini hanno grande flessibilità nell'interpretare correttamente segnali rumorosi o incompleti.

Alcuni linguaggi NON prevedono affatto che si scrivano le vocali!

Se si ha una "competenza" spesso si possono completare da soli i dettagli che non sono stati trasmessi in maniera completa (esempio di ridondanza semantica). Ad esempio alcuni telefonini sanno "scegliere" quando si compone un SMS quale è la sequenza di caratteri più "probabile" nella nostra lingua (esempio di ridondanza statistica).

Ni mzz dl cmmn d nstr vt
M rtrv pr n slv scr

Idea della compressione lossy

La regola è semplice: se "percettivamente" non è importante, buttalo via!

- **MP3:** applicano questa idea al caso del suono e della musica
- **JPEG :** applicano questa idea alle immagini fisse (still images)
- **MPEG, AVI, DVX, etc:** applicano questa idea alle sequenze di immagini (filmati)

Ovviamente una volta buttata via l'informazione non può essere ricostruita: si tratta di una compressione IRREVERIBILE.

Algoritmo lossy banale: Requantization

Si tratta molto semplicemente di una riduzione del numero di livelli disponibili in modo da risparmiare bit per pixel. La si realizza "dimenticando" n bit meno significativi per canale.

Es. RED: da 8 bit si conservano solo i 4 più significativi;
GREEN: da 8 bit si conservano solo i 6 più significativi;
BLUE: da 8 bit si conservano solo i 2 più significativi.

Si risparmia così il 50% dei bit inizialmente necessario. Inoltre se ci sono meno simboli la compressione LZW o Huffman è più efficiente.
Da sottolineare la grande perdita di qualità nelle immagini.

Compressione dello standard JPEG

Storia

JPEG è l'acronimo di "Joint Photographic Experts Group" (www.jpeg.org). Lo standard JPEG è stato sviluppato per la compressione di immagini.

Nel 1988, JPEG ha scelto uno schema di codifica adattativo basato sulla tecnica DCT (Discrete Cosine Transform)

Nel 1991 è stata presentata ufficialmente una proposta di standard che è stata approvata dai membri del consorzio ed è diventata uno standard ISO nel 1992 (ISO –International Standard Organization).

Passi fondamentali della codifica JPEG

- **Pre-processing:**
 - I. Color Transform (RGB → YC_bC_r);
 - II. Sottocampionamento della crominanza;
 - III. Suddivisione dell'immagine in sottoimmagini.
- **Trasformazione:**
 - I. Discrete Cosine Transform (DCT);
 - II. Quantization.
- **Codifica:**
 - I. DC Coefficient Encoding;
 - II. Zig-zag ordering of AC Coefficients;
 - III. Entropy Coding (Huffman).

Preprocessing (I): da RGB a YC_bC_r

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.229 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Y è il canale della luminanza, mentre C_b e C_r sono i canali della crominanza.

La crominanza è una proprietà dello spettro di un segnale televisivo che, unita alla luminanza, permette di ricostruire immagini a colori. La sua applicazione pratica è spesso nei sistemi che possono utilizzare terminali sia in bianco e nero sia a colori, quindi tipicamente la televisione.

Per permettere ai televisori in bianco e nero di visualizzare un segnale televisivo destinato ai televisori a colori RGB, invece che trasmettere per ogni linea questi tre colori primari le informazioni vengono trasmesse separatamente (video a componenti) secondo il seguente schema:

- i valori di luminanza (Y), che rappresentano l'intensità di luce complessiva dell'immagine (cioè la somma dei tre colori primari),
- i valori di differenza dal colore rosso (C_r),
- i valori di differenza dal colore blu (C_b).

dando vita ad un segnale YC_bC_r a volte detto impropriamente anche segnale YUV.

In questo modo un televisore in bianco e nero, pur ricevendo tutti e tre i segnali, utilizza solo il primo, quello di luminanza, ignorando gli altri due; al contrario, un televisore a colori utilizza i segnali del rosso e del blu trasmessi, e ne ricava il verde, sottraendo al segnale di luminanza le informazioni su rosso e blu.

Nelle seguenti formule, viene riassunto il modo di ottenere i tre colori a partire dalla luminanza e dalle due crominanze:

- $Y = \text{Blu} + \text{Rosso} + \text{Verde}$
- $C_b = \text{Blu} \cdot Y$
- $C_r = \text{Rosso} \cdot Y$

Si tratta di una trasformazione comunque reversibile.

Preprocessing (II): sottocampionamento della crominanza

Il modello YC_bC_r è un modello di colori che permette di avvantaggiarsi della "debolezza" del sistema visivo umano.

Poiché l'occhio umano è più sensibile alla luminanza che alla crominanza è possibile adottare tecniche di sottocampionamento della crominanza attuando cioè una forma di codifica di sorgente o compressione dati dell'immagine a colori da trasmettere abbassando il bit-rate (banda) complessivo.

JPEG prende TUTTE le informazioni sulla luminanza ma sceglie solo UN campione delle informazioni degli altri canali.

Si può "personalizzare" la compressione scegliendo 1 valore ogni 4 per C_b e C_r (tralasciando metà dei valori originari da comprimere) oppure scegliendo 2 valori ogni 4 per C_b e C_r (tralasciando un terzo dei valori originali da comprimere)

Questo passo è ovviamente con perdita di informazione, e quindi irreversibile.



Preprocessing (III): partizione della immagine

Per approfittare al meglio della ridondanza, dato che localmente è maggiore che sulla intera immagine, e anche per semplificare i calcoli, JPEG procede dividendo l'immagine in **quadrotti** 8×8 di 64 pixel non sovrapposti.

Quadrotti diversi subiranno una elaborazione differente: è qui l'origine del noto problema "quadrettatura" spesso visibile in ingrandimenti o stampe di immagini che sono state compresse con JPEG.

Prima della prossima fase (DCT), ai 64 pixel di ciascun blocco viene sottratta una quantità pari a 2^{n-1} , dove 2^n rappresenta il numero massimo di livelli di grigio dell'immagine. Se il blocco considerato presenta $256 = 2^8$ possibili livelli di grigio, a ciascun pixel di tale blocco verrà sottratto un offset pari a $128 = 2^7$. Con questo processo, noto come **shift dei livelli di grigio**, il grigio medio (128) diventa 0.

Esempio:

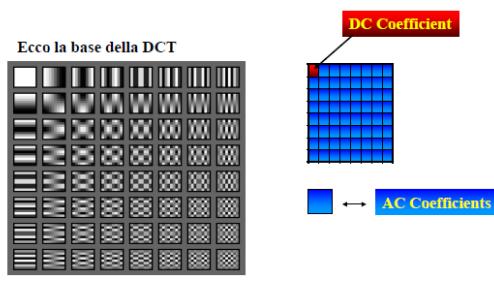
52	55	61	66	70	61	64	73		-76	-73	-67	-62	-58	-67	-64	-55
63	59	66	90	109	85	69	72		-65	-69	-62	-38	-19	-43	-59	-56
62	59	68	113	144	104	66	73		-66	-69	-60	-15	16	-24	-62	-55
63	58	71	122	154	106	70	69	→ -128	-65	-70	-57	-6	26	-22	-58	-59
67	61	68	104	126	88	68	70		-61	-67	-60	-24	-2	-40	-60	-58
79	65	60	70	77	68	58	75		-49	-63	-68	-58	-51	-60	-70	-53
85	71	64	59	55	61	65	83		-43	-57	-64	-69	-73	-67	-63	-45
87	79	69	68	65	76	78	94		-41	-49	-59	-60	-63	-52	-50	-34

Trasformazione (I): DCT

Il JPEG trasforma i blocchi quadrotti di 8x8 pixel secondo un algoritmo detto **Trasformata Discreta del Coseno** (Discrete Cosine Tranform, DCT). Si tratta di un algoritmo della famiglia delle trasformate di Fourier. È stato dimostrato che, statisticamente, tale trasformazione *decorrela* al massimo i dati permettendo maggiori rapporti di compressione nella fase successiva di codifica. Il nome non deve confondere: si tratta di una trasformazione del "vettore" di 64 pixel dalla base impulsiva (canonica) ad una più adatta alle immagini.

Come osservato, un'immagine di 8x8 pixel si può pensare come un vettore nello spazio a 64 dimensioni. Ogni immagine è quindi la somma pesata di 64 immagini impulsive (tutte nere tranne in un pixel di valore 1) ove i "pesi" rappresentano l'effettivo livello di luminosità di ogni

pixel. Tali immagini impulsive costituiscono una base, detta **base impulsiva** per le immagini. La "base impulsiva" non è l'unica base. La trasformata del coseno esprime l'immagine in un'altra base.



Ogni immagine 8x8 si ottiene moltiplicando ciascuna delle immagini a sinistra per un coefficiente e sommando tutte le immagini.
I coefficienti di tale somma sono i coefficienti della DCT.
Il coefficiente in alto a sinistra è un valore proporzionale al valor medio della luminanza dell'immagine. E' detto anche coefficiente DC

La formula della DCT per un blocco di dimensioni $N \times N$ (N=8 nel jpeg) è:

$$F(u, v) = \frac{2}{N} \left[\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} C(u)C(v)f(x, y) \cos \frac{(2x+1)u\pi}{2 * N} \cos \frac{(2y+1)v\pi}{2 * N} \right]$$

$$f(x, y) = \frac{2}{N} \left[\sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u)C(v)F(u, v) \cos \frac{(2x+1)u\pi}{2 * N} \cos \frac{(2y+1)v\pi}{2 * N} \right]$$

dove

$$C(u) = \frac{1}{\sqrt{2}} \quad \text{per} \quad u = 0 \quad C(u) = 1$$

$$C(v) = \frac{1}{\sqrt{2}} \quad \text{per} \quad v = 0 \quad C(v) = 1$$

Una implementazione diretta delle formule sopra richiede complessità $O(N^2)$. Esistono algoritmi "fast" per calcolare i coefficienti in $O(N \log(N))$ derivati dalla Fast Fourier Transform.

Es.	-76 -73 -67 -62 -58 -67 -64 -55 -65 -69 -62 -38 -19 -43 -59 -56 -66 -69 -60 -15 16 -24 -62 -55 -65 -70 -57 -6 26 -22 -58 -59 -61 -67 -60 -24 -2 -40 -60 -58 -49 -63 -68 -58 -51 -60 -70 -53 -43 -57 -64 -69 -73 -67 -63 -45 -41 -49 -59 -60 -63 -52 -50 -34	→ DCT	-415 -29 -62 25 55 -20 -1 3 7 -21 -62 9 11 -7 -6 6 -46 8 77 -25 -30 10 7 -5 -50 13 35 -15 -9 6 0 3 11 -8 -13 -2 -1 1 -4 1 -10 1 3 -3 -1 0 2 -1 -4 -1 2 -1 2 -3 1 -2 -1 -1 -1 -2 -1 -1 0 -1
-----	--	-------	---

Trasformazione (II): Quantizzazione

Un vantaggio in termini di simboli da usare (e quindi in termini di lunghezza dei codici Huffman) si ottiene se si riduce il numero di "livelli" su cui i coefficienti della DCT possono variare.

Tale operazione permette di rappresentare i diversi coefficienti incrementando il fattore di compressione, più precisamente avviene un processo di riduzione del numero di bit necessari per memorizzare un valore intero riducendone la precisione.

Si usa il seguente "formalismo" per la quantizzazione. Dato un **fattore di quantizzazione Q** e un **numero F** il valore $F_{\text{quantizzato}}$ si ottiene come:

$$F_{\text{quantizzato}} = \text{round}(F/Q)$$

Il valore ricostruito si ottiene moltiplicando $F_{\text{quantizzato}}$ per Q. Ovviamente la quantizzazione è un processo *irreversibile* (perdita di informazione)

Si è scoperto sperimentalmente che non è conveniente usare un unico fattore di quantizzazione per tutti i 64 coefficienti della DCT della luminanza, o per quantizzare i valori provenienti dalla DCT delle crominanze.

Si preferisce adottare per il coefficiente $F(i,j)$ un fattore di quantizzazione $Q(i,j)$ scelto a priori (fornito dallo standard) o scelto dall'utente (in questo caso la tabella di quantizzazione deve essere trasmessa assieme ai dati compressi per consentire una corretta ricostruzione). I fattori $Q(i,j)$ costituiscono la cosiddetta "*tabella di quantizzazione*".

Tabella di quantizzazione luminanza

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

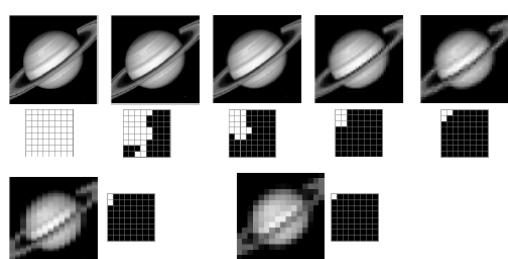
Tabella di quantizzazione crominanza

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

Si osservi che un fattore di compressione maggiore comporta una maggiore perdita di informazione.

L'utente del JPEG può scegliere il "grado" di quantizzazione da adottare fornendo un "quality factor" QF che va da 1 a 100. La tabella di quantizzazione adottata sarà una copia delle tabelle sopra i cui elementi sono divisi per QF.

Maggiore il QF, minore i fattori di quantizzazione e minore la perdita di informazioni.



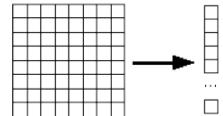
A sinistra un'immagine che spiega gli effetti della quantizzazione: I quadretti neri rappresentano i coefficienti DCT che la quantizzazione ha portato a zero per ogni blocco 8x8 della immagine di Saturno.

Es.

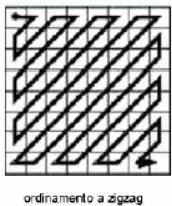
-415 -29 -62 25 55 -20 -1 3	16 11 10 16 24 40 51 61	-26 -3 -6 2 2 0 0 0
7 -21 -62 9 11 -7 -6 6	12 12 14 19 26 58 60 55	1 -2 -4 0 0 0 0 0
-46 8 77 -25 -30 10 7 -5	14 13 16 24 40 57 69 56	-3 1 5 -1 -1 0 0 0
-50 13 35 -15 -9 6 0 3	14 17 22 29 51 87 80 62	-4 1 2 -1 0 0 0 0
11 -8 -13 -2 -1 1 -4 1	18 22 37 56 68 109 103 77	1 0 0 0 0 0 0 0
-10 1 3 -3 -1 0 2 -1	24 35 55 64 81 104 113 92	0 0 0 0 0 0 0 0
-4 -1 2 -1 2 -3 1 -2	49 64 78 87 103 121 120 101	0 0 0 0 0 0 0 0
-1 -1 -1 -2 -1 -1 0 -1	72 92 95 98 112 100 103 99	0 0 0 0 0 0 0 0

Codifica (I): Codifica dei coefficienti DC

I coefficienti DC della DCT (cioè i valori "medi" dei blocchi 8x8) vengono codificati come uno stream a parte. Tutti gli altri coefficienti AC vengono riordinati in un vettore 63x1 seguendo l'ordinamento "a serpentina" (per creare lunghe run di zeri) e codificati in un altro stream.



Codifica (II): Ordinamento a Zig-Zag



0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63

Gli indici che determinano l'ordinamento a zig-zag dei coefficienti quantizzati

Codifica (III): Codifica ad Entropia

A questo punto si hanno due differenti codifiche. I coefficienti DC, cioè quelli che stanno nella posizione (1,1) del blocco 8x8, sono codificati usando una codifica differenziale; I coefficienti AC, cioè tutti gli altri del blocco, sono codificati usando una codifica run-length.

➤ Codifica differenziale dei coefficienti DC

Se la sequenza dei valori varia lentamente, invece di registrare i valori è sufficiente ricordarsi del valore iniziale e delle differenze successive.

Es. Data la serie 134, 137, 135, 128, 130, 134, 112, ...

Ricordo il valore iniziale 134 e poi la sequenza delle differenze successive: -3, +2, -2, -4, 22, ...

Si dimostra sperimentalmente che per le immagini la sequenza delle differenze ha una entropia minore di quella dei valori originali e quindi richiede meno bit per essere memorizzata.

I coefficienti del blocco 8x8 messi in sequenza sono:

-26 -3 1 -3 -2 -6 2 -4 1 -4 1 1 5 0 2 0 0 -1 2 0 0 0 0 0 -1 -1 EOB

Il coefficiente DC è il primo e vale -26. Assumendo che il coefficiente DC del blocco successivo sia -17, otterremo l'evento

$$\Delta = -26 - (-17) = -9$$

Nella **tavola delle categorie** (a destra), -9 sta nella posizione n=SSSS=4, dove SSSS è la categoria, e Δ è la differenza tra due coefficienti DC (evento).

Il valore n=SSSS=4 ha come codice base 101.

La corrispondenza tra il valore e il codice è fissata dalla **tavella dei codici di Huffman** (in basso a sinistra) che varia in base al fatto che stiamo trattando la luminanza o la crominanza.

A questo punto occorre completare il codice. Per fare ciò si usa la seguente regola:

- Se $\Delta > 0$ allora i bit da aggiungere sono gli n bit meno significativi del valore in binario di Δ
- Se $\Delta < 0$ allora i bit da aggiungere sono gli n bit meno significativi del valore in binario di Δ ai quali bisogna sottrarre il valore 1
- Se $\Delta = 0$ anche SSSS è uguale a zero, pertanto non viene aggiunto nessun bit.

Nell'esempio considerato i quattro bit meno significativi del valore in binario di Δ (-9) sono 0111; essendo $\Delta < 0$ si sottrae il valore 1 e si ottengono i quattro bit 0110 che completano il codice base trovato in precedenza (101).

Il codice completo del coefficiente DC è 1010110.

➤ Codifica dei coefficienti AC

Dalla sequenza si elimina il primo coefficiente e si passa alla codifica di tutti gli altri.

Poiché i coefficienti quantizzati AC sono spessissimo nulli, si usa una trasformazione in **skip-value**, cioè data una sequenza di valori, si memorizza il numero degli zeri seguito dal primo valore non zero che si incontra.

SSSS	Δ
0	0
1	-1, 1
2	-3 -2, 2 3
3	-7 ... -4, 4 ... 7
4	-15 ... -8, 8 ... 15
5	-31 ... -16, 16 ... 31
6	-63 ... -32, 32 ... 63
7	-127 ... -64, 64 ... 127
8	-255 ... -128, 128 ... 255
9	-511 ... -256, 256 ... 511
10	-1023 ... -512, 512 ... 1023
11	-2047 ... -1024, 1024 ... 2047

SSSS	Codice base
0	010
1	011
2	100
3	00
4	101
5	110
6	1110
7	11110
8	111110
9	1111110
10	11111110
11	111111110

Es. Si debba codificare 0,0,0,0,11,0,0,0,3,0,0,0,0,0,0,0,0,12,17...
Memorizzo: (4,11),(3,3),(8,12),(0,17),...

Nel nostro caso -26 -3 1 -3 -2 -6 2 -4 1 -4 1 1 5 0 2 0 0 -1 2 0 0 0 0 0 -1 -1 EOB sarebbe: (0,-3), (0,1), (0,-3), (0,-2), (0,-6), (0,2), (0,-4), (0,1), (0,-4), (0,1), (0,1), (0,5), (1,2), (2,-1), (0,2), (5,-1), (0,-1) ...

La prima coppia da codificare è (0,-3). La categoria -3 ha come SSSS il valore 2. La classe dell'evento è espressa mediante una coppia del tipo (run, categoria).

Nel nostro caso abbiamo (0,2), al quale corrisponde il codice base 01, tale codice sarà completato dall'aggiunta di un numero di bit n pari alla categoria (n = SSSS).

Gli n bit che completano il codice base sono scelti con lo stesso criterio enunciato per la codifica dei coefficienti DC, in base al valore v del coefficiente AC ($v > 0$, $v < 0$, $v = 0$).

Nell'esempio considerato i due bit meno significativi del valore in binario del coefficiente AC v (-3) sono 01, essendo $v < 0$ si sottrae il valore 1 e si ottengono i due bit 00 che completano il codice base trovato in precedenza (01).

Il codice completo è quindi 0100.

Immagine finale compressa

La sequenza finale codificata sarà

101010 0100 001 0100 0101 100001 0110 100011 001 100011 001 001 100101 11100110 110110 0110 11110100 000 1010
Dove gli spazi sono inseriti solo per migliorare la leggibilità.

Ricostruzione della risoluzione originale

Si deve tornare indietro "ricostruendo" i dati originali (o le loro approssimazioni per i passi irreversibili). Esistono diverse "strategie" per la ricostruzione in modo da abilitare la ricostruzione progressiva, gerarchica o lossless.

Attenzione, il Jpeg è lossy! Quindi il blocco ricostruito è diverso da quello in input.

52	55	61	66	70	61	64	73
63	59	66	90	109	85	69	72
62	59	68	113	144	104	66	73
63	58	71	122	154	106	70	69
67	61	68	104	126	88	68	70
79	65	60	70	77	68	58	75
85	71	64	59	55	61	65	83
87	79	69	68	65	76	78	94

Blocco di input

58	64	67	64	59	62	70	78
56	55	67	89	98	88	74	69
60	50	70	119	141	116	80	64
69	51	71	128	149	115	77	68
74	53	64	105	115	84	65	72
76	57	56	74	75	57	57	74
83	69	59	60	61	61	67	78
93	81	67	62	69	80	84	84

Blocco ricostruito

Es. Fattori di qualità indicati sotto le immagini



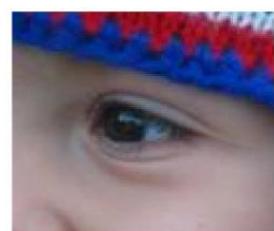
1



16



50



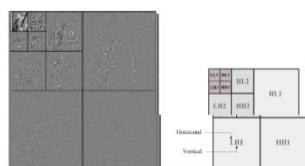
100

Immagini "grafiche" con pochi colori e con testi non sono compresse con buona qualità dal JPEG!

Inoltre, per il WEB, JPEG non gestisce la trasparenza (GIF e PNG lo fanno).

Nuovi Standard: JPEG2000

Sostituisce la DCT con le wavelets. Alloca più bit nelle zone con più informazione e permette il controllo esplicito di tale allocazione. Raggiunge rapporti di compressione più elevati. Non è stato un successo commerciale (inerzia tecnologica)



SSSS	Coefficiente AC
1	-1, 1
2	-3 -2, 2 3
3	-7 ... -4, 4 ... 7
4	-15 ... -8, 8 ... 15
5	-31 ... -16, 16 ... 31
6	-63 ... -32, 32 ... 63
7	-127 ... -64, 64 ... 127
8	-255 ... -128, 128 ... 255
9	-511 ... -256, 256 ... 511
A	-1023 ... -512, 512 ... 1023

(run, category)	Codice base	Lunghezza codice completo
(0, 0)	1010 (= EOB)	4
(0, 1)	00	3
(0, 2)	01	4
(0, 3)	100	6
....
(F, 0)	111111110111	12
....
(F, A)	11111111111111110	26

Matlab

Cos'è Matlab

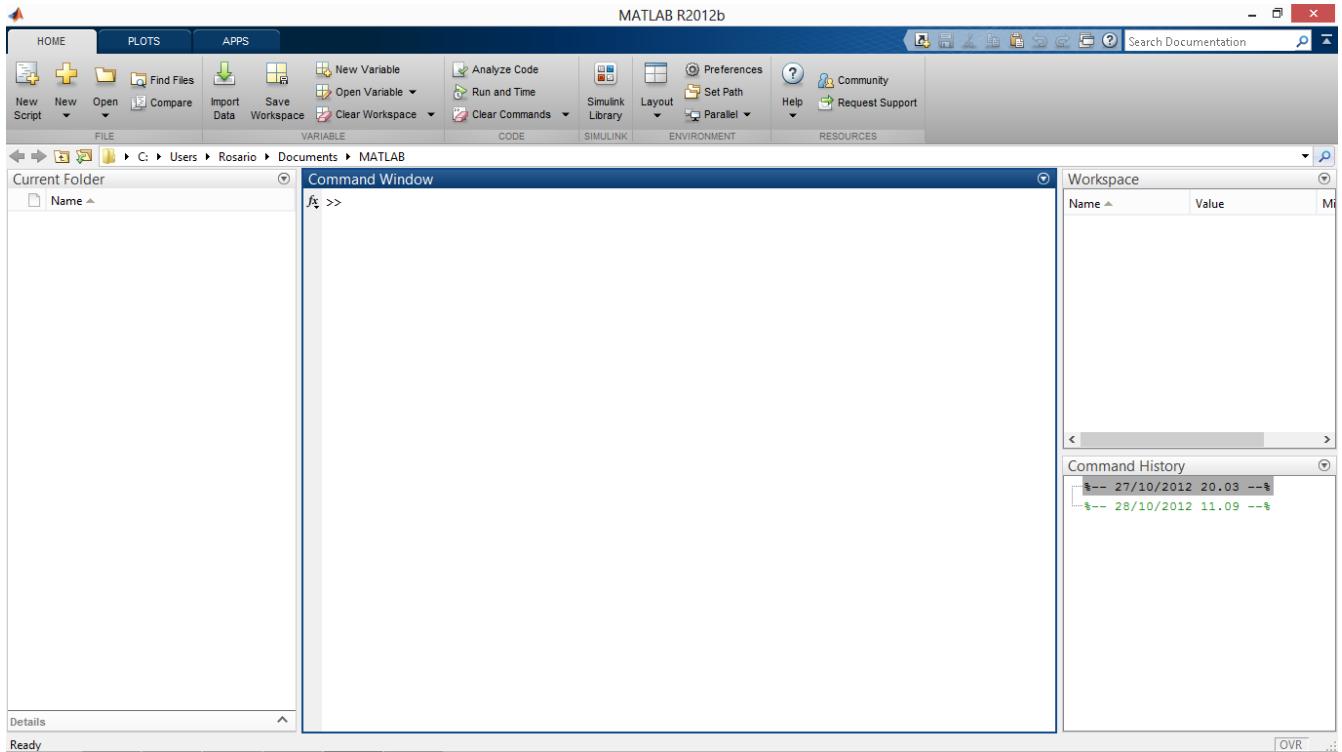
Matlab (abbreviazione di **M**atrix **L**aboratory) è un ambiente per il calcolo numerico, visualizzazione grafica e l'analisi statistica che comprende anche l'omonimo linguaggio di programmazione ad alte prestazioni creato dalla *MathWorks*. MATLAB consente di manipolare matrici, visualizzare funzioni e dati, implementare algoritmi, creare interfacce utente, e interfacciarsi con altri programmi.

Esso è implementato in un potente ambiente di lavoro ad interfaccia grafica, avente una struttura modulare che ne rende notevolmente espandibili le funzionalità. È possibile infatti installare moduli aggiuntivi, detti **Toolbox**, che forniscano caratteristiche specifiche, non presenti nel set di funzioni originali di Matlab.

MATLAB è usato da milioni di persone nell'industria e nelle università per via dei suoi numerosi tool a supporto dei più disparati campi di studio applicati e funziona su diversi sistemi operativi, tra cui Windows, Mac OS, GNU/Linux e Unix.

Ambiente di lavoro

L'ambiente di lavoro è suddiviso in quattro aree principali: *Command Window*, *Workspace*, *Current Folder*, *Command History*.



Screenshot di Matlab 2012

- **Command Window:** Finestra dove andremo a digitare i comandi supportati e visualizzare in tempo reale i risultati. In sostanza è la shell testuale dell'ambiente di lavoro ed è senz'altro lo strumento più potente che abbiamo a disposizione per usufruire in modo completo della enorme mole di programmi presenti in Matlab.
- **Workspace:** Potente strumento di monitoraggio delle variabili utilizzate nel corso delle operazioni effettuate dall'utente. Possiamo ottenere informazioni sul tipo e sull'occupazione di memoria di una certa variabile e addirittura accedere per via grafica direttamente al contenuto della variabile stessa.
- **Current Folder:** Tramite questa cartella possiamo spostarci tra le cartelle del nostro Hard Disk come faremmo con un qualsiasi File Manager. Di default essa visualizza il contenuto della cartella /work contenuta nella directory principale di Matlab
- **Command History:** Lista di tutti i comandi digitati, la "storia" dei comandi per l'appunto, con utilissime funzioni di copia e incolla.

Script (M-File)

Un M-File è un file in cui è possibile salvare intere sequenze di comandi, anche molto lunghe (come dei metodi, per fare un paragone con Java). Sul nostro hard disk un M-File presenterà un'estensione ".m".

È possibile dunque eseguire in modo rapido sequenze di azioni semplicemente lanciando l'M-File opportunamente creato. Per farlo basta spostarsi nella cartella che lo contiene mediante la shell testuale, attraverso il comando "cd", e digitare il nome completo dell'M-File, senza l'estensione ".m", o semplicemente fare doppio click su esso per aprirlo in Matlab.

Poiché Matlab offre funzionalità di programmazione un M-File può contenere anche dei veri e propri programmi e non solo semplici "liste" di comandi indipendenti. Possiamo in questo modo aggiungere altre funzioni, anche complesse, a quelle che già ci vengono offerte in partenza, allo scopo di risolvere problemi specifici.

Function

Una **function** di Matlab è una lista di comandi che necessita di variabili di input per essere eseguita e restituisce variabili di output. Una funzione è contenuta in un file ".m" che ha lo stesso nome della function stessa; il file che definisce la function deve cominciare con

```
function [argomento_output] = nome_funzione (argomenti_input)
```

Es.
function f = my_function(x);
f = x^3-2*sin(x)+1;
end;

Sarà dunque possibile utilizzare il comando `my_function(x)` come fosse qualunque altro comando Matlab. Le linee di commento che seguono l'intestazione costituiscono l'help della function e vengono visualizzate qualora di esegua `>>help function`. Esse sono precedute dal carattere %. Tutte le variabili definite internamente alla function sono locali.

Rudimenti di Matlab

In allegato a questo documento troverete degli Script da visualizzare con Matlab contenenti tutto ciò che serve per programmare al meglio con questo linguaggio.

Scalari, vettori e matrici

```
%% NAVIGAZIONE TRA CARTELLE

%Spostamento tra cartelle
cd nome_cartella
cd .. %tornare su
cd C:\

%Elenco files
dir      %eleco dei files della cartella

%Help
help      %sescrizione sintetica online delle funzionalità
help nome_cartella
help nome_comando

%Commenti
%      %riga di commento in Matlab
CTRL+R  %commenta tutte le righe selezionate
CTRL+T  %decommenta tutte le righe selezionate

%% SCALARI
% Una costante è una matrice 1x1 in Matlab

%Variabili predefinite:
i,j          %unità immaginaria in un numero complesso
pi           %approssimazione di pi greco, ?
eps          %precisione di macchina del pc che si sta utilizzando
realmax      %è il massimo numero reale positivo rappresentabile
realmin      %è il minimo numero reale positivo rappresentabile
inf           %è un numero maggiore di realmax, ?, infinito
version       %restituisce la versione di MATLAB utilizzata
computer     %restituisce il tipo di computer utilizzato. Esempio: PCWIN
ans           %ultimo risultato dell'ultima operazione eseguita
NaN           %"Not a number", indica il risultato di una forma matematica indeterminata
(es.?/?)

%Dichiarazioni di variabili
var1=1        %dichiarazione di una variabile (di default è double)
var2='abc'    %caratteri e stringhe dichiarate tra i "doppi apici"
var3=1234;    %il ";" finale non fa visualizzare l'istruzione ma la memorizza
```

```

%Reset
clear nome_variabile %cancellare una determinata variabile
clear %reset di tutta la workspace
close all %chiudere tutte le finestre aperte

%Visualizzare variabili
who %visualizzare variabili senza descrizione approfondita
whos %visualizzare variabili con descrizione approfondita

%Operazioni avanzate per gli scalari
sqrt(x) %radice quadrata di x
log(x) %logaritmo naturale di x
help elfun %lista completa di tutte le funzioni matematiche

%% VETTORI

%Tipi
vettR=[2 8 10 7] %vettore riga (1xn)
vettC=[2;8;10;7] %vettore colonna (mx1)

%Dichiarazione con intervalli
vett=[inizio:fine] %es. v1=[1:10] genera v1=[1 2 3 4 5 6 7 8 9 10]
vett=[inizio:incremento:fine] %es. v2=[-1 : 0.2 : 1] genera v2=[-1 -0.8 -0.6 -0.4 -
0.2 0 0.2 0.4 0.6 0.8 1]

%Selezione
v(k) %seleziona un elemento del vettore, dove k è l'indice che conta a partire da 1

%% MATRICI

%Dichiarazione
M1=[1 2 3 4;5 6 7 8;9 10 11 12]
M2=[13,14,15;16,17,18]

%Dichiarazione per parti
M(1,1:3)=[a1 b1 c1]; %prima riga, dalla posizione 1 a 3 i valori tra []
M(2,1:3)=[a2 b2 c2];
M(3,1:3)=[a3 b3 c3];

%Dichiarazione con intervalli
M=[inizio:fine]
M=[inizio:incremento:fine] %es. M=[1:2:7;4:2:10;7:2:13] genera M=[1 3 5 7; 4 6 8 10;7
9 11 13]

%Selezione
M(l,k) %dove l e k sono gli indici riga e colonna della matrice

%Estrazione
M(l,:) %estrae l'intera riga l-esima
M(:,k) %estrae l'intera colonna k-esima
M(:) %estrae l'intera matrice

%Eliminazione
M(:,k)=[] %elimina la colonna k-esima

%% OPERAZIONI CON LE MATRICI
%Devono essere rispettate le regole relative all'algebra delle radici

%Somma e differenza
A=[1 2;3 4];
B=[5 6;7 8];
C=A+B;
>> C=[6 8;10 12]

```

```

%Prodotto righe per colonne
A=[1 2;3 4];
B=[5 6 7;8 9 10];
C=A*B;
>> C=[21 24 27;17 54 61]

%Prodotto puntuale
A=[1 2;3 4];
B=[5 6;8 9];
C=A.*B; %ogni elemento della matrice A con un elemento della matrice B nella stessa posizione
>> C=[5 12;24 36]

%Divisione
A/B = A*inv(B) %divisione a destra
A\B = inv(B)*A %divisione a sinistra

%Size
A=[1 2 3;4 5 6]
size(A) % restituisce un vettore composto dal numero di righe e di colonne della matrice A e i livelli di colore
>> ans = 2 3 3

m=size(I,1) %assegnare a m il numero di righe
n=size(I,2) %assegnare a n il numero di colonne

%Trasposizione
A=[1 2;3 4];
B=A';
>> B=[1 3;2 4]

%Sum
A=[4 5;7 6;8 9]
sum(A) %restituisce un vettore riga composto dalla somma di ogni colonna di A
>> ans = 19 20

%Elevamento a potenza
A=[2 4;5 6];
B=A^2 %se la matrice è quadrata, restituisce il prodotto riga per colonna di A per se stessa n volte
>> B=[24 32;40 56]

C=[3 4;5 6;7 8]
D=C.^2 %elevazione a potenza dei singoli elementi della matrice
>> D=[9 16;25 36;49 64]

%Determinante
det(A)

%Polinomio caratteristico
poly(A)

%Autovalori
eig(A)

%Diagonale principale
A=[4 5 6;8 9 10;11 56 33]
diag(A) %si ottiene un vettore colonna composto da tutti gli elementi della diagonale principale di A.
>> ans = [4;9;23]

%Somma degli elementi della diagonale principale
trace(A)

%Ordine delle colonne invertito

```

```

fliplr(A)

%Inversa
inv(A)

%Complemento
imcomplement(I);

```

Grafici

```

%% GRAFICI LINEARI

plot(x);      %produce un grafico lineare (x può essere uno scalare, vettore o matrice)
plot(x,y);    %produce un grafico lineare di y in funzione di x
plot(x,'m s c');    %dove marcatore(+,*,,x) stile linea(-,--, :) e colore
subplot(m,n,p);    %grafi nella stessa finestra in una matrice mxnxposizione_grafico
%|ES|   subplot(121),imshow(I); subplot(122),imshow(R);

%% GRAFICI 3D

mesh(x)      %produce un grafico 3D di una matrice di elementi

%% NOTAZIONI GRAFICHE
title('testo')      %titolo al grafico
xlabel('testo')      %nome all'asse delle ascisse
ylabel('testo')      %nome all'asse delle ordinate
figure(n)           %creare tante finestre quanto assegniamo al valore n

```

Manipolazione di matrici

```

%% LETTURA DI UNA IMMAGINE

I=imread(filename,estensione);      %legge un'immagine in scala di grigi o a colore
I=imread('percorso',estensione);    %restituisce una matrice mxn se a scala di grigi,
mxnx3 se a colori RGB, mxnx4 se a colori CMYK (file *.tiff)

%gli elementi della matrice sono valori da 0 a 255 e di tipo uint8

%% VISUALIZZAZIONE DI UNA IMMAGINE

imshow(I)                  %visualizzare l'immagine I
rgb2gray(I)                %conversione di un'immagine a colori in scala di grigi
im2bw(I)                   %conversione di un'immagine in bianco e nero
im2bw(I, level)            %conversione di un'immagine in bianco e nero), tutti i pixel
con intensità superiori a level vengono posti a 1, quelli più piccoli a 0
imshow(I,[low high])       %con immagini a scala di grigi, fissa il range dei valori da
visualizzare; se val<low -> nero; se val>high -> bianco
imshow(I,map)               %visualizza un'immagine indicizzata on relativa mappa di
colore
image(A)                   %visualizzare l'immagine I partendo dalla matrice
image(x,y,A);              %visualizzare l'immagine I con intervalli x e y
imagesc(A);                %visualizzare l'immagine I con la mappa di colore

%Decomposizione immagine
R = I(:,:,1);             %estrarre il canale R di un'immagine RGB
G = I(:,:,2);             %estrarre il canale G di un'immagine RGB
B = I(:,:,3);             %estrarre il canale B di un'immagine RGB
figure(2),imshow(R),title('rosso');
figure(3),imshow(G),title('verde');
figure(4),imshow(B),title('blu');
figure,imshow(uint8([R,G,B]));  %mostrare in serie la sequenza dei colori
dell'immagine
figure,imshow(uint8([R;G;B])); %mostrare in colonna la sequenza dei colori
dell'immagine
figure,imshow(uint8([I(:,:,1),I(:,:,2),I(:,:,3)])); %visualizzare immagine in sequenza
su unica finestra

% Scorrere la matrice dell'immagine e stamparla a video
for i=1:m      %il ciclo inizializza i ad 1, ed arriva fino a m (una variabile da noi

```

```

creata contenente dimensione delle righe)
for j=1:n
    I(i,j)
end
end

mesh((I(:,:,1))); %mostrare il grafico dei colori

%% SCRIVERE UN FILE IMMAGINE

imwrite(I,filename,estensione)      %salvare su disco un file immagine
imwrite(I,map,filename,estensione); %salvare su disco un file immagine con la relativa
mappa di colore

%% MAPPE DI COLORE
%Sono matrici mx3 di valori tra [0 1] che specificano i valori RGB

colormap(map);                      %utilizzare la mappa di colore specificata da map
colormap('default');                %utilizzare la mappa corrente come default
cmap = colormap;                   %restituisce la mappa di colore dell'immagine nella matrice
cmap
rgbplot(cmap)                      %disegnare manualmente una mappa di colori
colormapeditor                      %editare una mappa di colore esistente

%% MODIFICARE LUMINOSITA'
%Per applicarla bisogna modificare la mappa di colore dell'immagine
%Mappa di colore schiarita se 0<beta<1
%Mappa di colore scurita se -1<beta<0

brighten(beta);
newmap = brighten(beta);

%% MODIFICARE IL CONTRASTO
%Per applicarla bisogna modificare la mappa di colore dell'immagine

cmap = contrast(X);
cmap = contrast(X,m);

%% MODIFICARE L'INTENSITA'

J = imadjust(I);      %crea una nuova immagine J tale che l'1% dei dati è saturato alle
basse ed alte intensità; il risultato è un aumento del contrasto dell'immagine
J = imadjust(I,[low_in; high_in],[low_out; high_out]); %intervallo dei valori
dell'immagine di partenza
J = imadjust(I,[low_in; high_in],[low_out; high_out],gamma); %gamma descrive la curva
di saturazione; se 0<gamma<1 saturazione elevata (risultato + luminoso), se gamma>1
valori bassi (risultato + scuro)
newmap = imadjust(map,[low in; high in],[low out; high out],gamma);

```

Operazioni affini

```

%% TRASLAZIONE

tx=30;  %valore x da traslare
ty=20;  %valore y da traslare

for v=1:m
    for w=1:n
        x=v+tx;
        y=w+ty;
        O(x,y)=I(v,w);
    end
end

imshow(O),title('output traslato');

```

```

%% RIDIMENTONAMENTO

B = imresize(I,scale); %restituisce un'immagine B che _e scale volte
1'immagine I (I<0.9 più piccola, I>1.1 più grande)
B = imresize(I,[mrows ncols]); %restituisce l'immagine A rapportata a mrows righe
e ncols colonne
[Y newmap] = imresize(X,map,scale);
[...] = imresize(..., method); %su method ci sta l'interpolazione

Metodi dell'interpolazione:
'nearest';
'bilinear';
'bicubic';
'box';
'triangle';
'cubic';
'lanczos2';

%% ROTAZIONE

theta=25;

A=[cosd(theta) sind(theta) 0; -sind(theta) cosd(theta) 0; 0 0 1];

for v=1:m
    for w=1:n
        R=[v w 1]*A;
        R=ceil(R);
        if R(1,1)>0 & R(1,2)>0
            O(R(1,1),R(1,2))=I(v,w);
        end
    end
end

figure,imshow(uint8(O),[]),title('output RUOTATO');

% é inoltre possibile ruotare un'immagine attarverso i seguenti comandi

B = imrotate(I,angle); %angle è l'angolo di rotazione
B = imrotate(I,angle,'method'); %method è l'interpolazione che può essere
'nearest', 'bilinear' o 'bicubic'
B = imrotate(I,angle,'method',bbox); %bbox specifica la dimensione dell'immagine
restituita B e può assumere i valori 'crop' (B _e della stessa dimensione di I) e
'loose' (ende l'immagine B grande quanto basta a contenere tutta l'immagine ruotata, è
di default)

%% ROTAZIONE E TRASLAZIONE

cx=2;
cy=2;
tx=30;
ty=20;
theta=25;

T=[1 0 0; 0 1 0; tx ty 1];
R=[cosd(theta) sind(theta) 0; -sind(theta) cosd(theta) 0; 0 0 1];
S=[cx 0 0; 0 cy 0; 0 0 1];

A=R*T*S;

O=zeros(m,n); %matrice di zeri

for x=1:m
    for y=1:n

```

```

R=[x y 1]*inv(A);
R=ceil(R);
if R(1,1)>0 & R(1,2)>0 & R(1,1)<=m & R(1,2)<=n
    O(x,y)=I(R(1,1),R(1,2));
end
end
end

figure,imshow(uint8(O)),title('output RUOTATO E TRASLATO');

%% ESTRAZIONE DI UNA PORZIONE DI IMMAGINE

I = imcrop; %seleziona con il mouse l'immagine da estrarre nella finestra attiva
I2 = imcrop(I);
X2 = imcrop(X,map);
I = imcrop(h);
I2 = imcrop(I,rect);
X2 = imcrop(X,map,rect);

%è comunque possibile indicare quale porzione estrarre con la variabile rect, che è il vettore di 4 elementi: [x_min y_min width height].
I2 = imcrop(I,[75 68 130 112]);

```

Istrogramma, equalizzazione ed estrazione contorni

```

%% ISTOGRAMMA

imhist(I); %calcola l'istogramma di I (se una matrice deve essere convertita in scala di grigi)
imhist(I, n); %calcola l'istogramma di I su n punti
imhist(X, map);
[counts,x] = imhist(...);
J = histeq(I, hgram); %equalizza l'istogramma della matrice I con hgram (istogramma di rigerimento)

%% ESTRAZIONE CONTORNI

B = bwboundaries(BW); %dove BW è un'immagine binaria e restituisce un array di celle, dove ogni cella è una matrice qx2, ovvero le q coppie di punti che formano il contorno dell'oggetto
J = edge(I); %metodo alternativo che usa gli edge (geometria dell'immagine)
J = edge(I,'method',tresh); %vengono ignorati i contorni che non superano la soglia tresh, i metodi supportati sono: sobel(di default), roberts, laplaciano gaussiano, zero-cross, canny
Y = diff(I); %estrazione tramite derivata numerica che evidenzia discontinuità tra bordo e sfondo

%% VALORE DI UN PIXEL

P = impixel(I); %permette di conoscere il valore RGB di un pixel cliccando col mouse sul punto d'interesse
P = impixel(X,map);

```

*Altri codici e funzioni Matlab possono essere trovati online