

Laboratorio di multimedialità

A.A. 2024-25
Andrea Macale

Indice

I Immagini	1
1 Elaborazione delle immagini	2
1.1 Sistema visivo umano	2
1.1.1 Cornea e coroide: due membrane come protezione	2
1.1.2 Retina: la membrana per la vista	3
1.1.3 Formazione dell'immagine	3
1.2 Rappresentazione delle immagini	4
1.2.1 Intensità di un'immagine	4
1.2.2 Digitalizzazione delle immagini	4
1.3 Manipolazione delle immagini	5
1.3.1 Manipolazioni principali	6
1.3.2 Funzione di trasformazione lineare a tratti	9
1.3.3 Equalizzazione dell'istogramma	10
2 Filtri nel dominio spaziale	13
2.1 Filtri di smussamento	14
2.1.1 Filtri di smussamento lineari	14
2.1.2 Filtri di smussamento non lineari	15
2.2 Filtri di nitidezza	16
2.2.1 Il Laplaciano	16
2.2.2 Filtri di contrasto	17
2.2.3 Filtri di rilevamento dei bordi	18
3 Filtri nel dominio della frequenza	21
3.1 Dominio della frequenza	21
3.1.1 Trasformata di Fourier in due dimensioni	22
3.1.2 Proprietà della trasformata di Fourier	22
3.1.3 Teorema della convoluzione e applicazione al filtraggio	23
3.2 Filtro di Notch	23
3.3 Filtri passa-basso	24
3.3.1 Filtro passa-basso ideale	24
3.3.2 Filtro passa-basso di Butterworth	25
3.3.3 Filtro passa-basso Gaussiano	25
3.4 Filtri passa-alto	26
3.4.1 Filtro passa-alto ideale, di Butterworth e Gaussiano	26
3.4.2 Il Laplaciano, i filtri di contrasto e l'high-boost filtering	27
4 Trasformata Wavelet	28
4.1 Limiti della trasformata di Fourier e possibile soluzione	28
4.1.1 Caratteristiche di un'immagine	28
4.1.2 Problemi della trasformata di Fourier	29
4.1.3 Una soluzione temporanea: la trasformata di Fourier a breve termine	29
4.2 Introduzione alla trasformata Wavelet	30

4.2.1	Determinazione della trasformata Wavelet	31
4.2.2	Proprietà	31
4.3	Analisi multirisoluzione	32
4.3.1	Creazione delle piramide dell'immagine	33
4.3.2	Rappresentazione walet e condizioni	34
4.4	Dalla trasformata di Haar alla trasformata Wavelet 2D	34
4.4.1	Trasformata Haar	34
4.4.2	Codifica di sottobanda	35
4.4.3	Vantaggi della trasformata di Haar e trasformata Wavelet 2D	36
4.4.4	Elaborazione dell'immagini nel dominio wavelet	36
5	Processo di compressione delle immagini	37
5.1	Ridondanza dei dati	37
5.1.1	Ridondanza di codifica	38
5.1.2	Ridondanza interpixel	38
5.1.3	Ridondanza psicovisiva	39
5.2	Teoria dell'informazione	39
5.2.1	Concetti base di informazione	40
5.2.2	Entropia	40
5.2.3	Criteri di fedeltà	40
5.3	Modello di compressione delle immagini	41
5.3.1	Codifica di Huffmann	41
5.3.2	Codifica aritmetica	42
5.3.3	Codifica RLC	43
5.3.4	Codifica di Lempel Ziv	43
5.3.5	Bit-plane coding	43
5.3.6	Standard di compressione lossy	43
6	Formato JPEG	44
6.1	Prima versione di JPEG	44
6.1.1	Fase 1: conversione da RGB a YCbCr	45
6.1.2	Fase 2: trasformata DCT	45
6.1.3	Fase 3: quantizzazione	45
6.1.4	Fase 4: pattern zig-zag	45
6.1.5	Fase 5: Codifica di entropia	46
6.2	JPEG2000	46
6.2.1	Fase 1: preprocessing	48
6.2.2	Fase 2: discrete Wavelet transform	48
6.2.3	Fase 3: codifica di blocchi	48
6.2.4	Fase 4: quantizzazione	49
6.2.5	Fase 5: EBCOT	49
6.2.6	Fase 6: codifica aritmetica adattiva	50
6.2.7	Fase 7: strati di qualità	50
7	Fondamenti della percezione della profondità	51
7.1	Indicatori monoculari di profondità statici	52
7.1.1	Occlusioni	52
7.1.2	Dimensioni relative	53
II	Video	54
8	Creazione dei video	55

Elenco delle figure

1	Struttura dell'occhio umano	2
2	Coni e bastoncelli	3
3	Illusione del mais dolce (a sinistra) e la griglia di Hermann (a destra)	4
4	Esempi di possibili manipolazioni di immagini	5
5	Possibili manipolazioni delle immagini	6
6	Immagine originale (a sinistra) e immagine in negativo (a destra)	7
7	Esempio di due immagini a cui sono state applicate con r differenti	7
8	Realizzazione della correzione gamma sui monitor	8
9	Calibrazione del gamma (azzurro la gamma dell'immagine, viola la gamma del display e la rossa la gamma complessiva)	8
10	Definizione analitica e grafico del contrast stretching	9
11	Definizione analitica e grafico del thresholding	9
12	Gray-level slicing a sfondo costante (a sinistra) e a sfondo invariato (a destra)	10
13	Esempio di istogramma di un'immagine 4x4 con livello di grigio [0, 9]	11
14	Calcolo dell'istogramma normalizzato	12
15	Esempio di istogramma di un'immagine equalizzata	12
16	Filtro media e gaussiano	14
17	Rimozione del rumore sale e pepe con un filtro mediano (sinistra originale e destra modificata)	15
18	Applicazione del Laplaciano (sinistra originale e destra modificata)	17
19	Originale, Laplaciano filtro di contrasto e high-boost filtering da in alto a sinistra in senso orario	18
20	Immagine originale (a sinistra) e immagine con i bordi rilevati con il rilevatore di Canny (a destra)	20
21	Rappresentazione della trasformata di Fourier	21
22	Immagine nel dominio spaziale (a sinistra) ed in frequenza (a destra)	22
23	Filtraggio nel dominio della frequenza	23
24	Immagine originale (a sinistra) e immagine filtrata con un filtro Notch (a destra)	23
25	Funzionamento di un filtro passo-basso ideale	24
26	Da sinistra: immagine originale, LPF ideale $D_0 = 8$ e LPF ideale $D_0 = 16$	24
27	Vari ordini del filtro passa-basso di Butterworth	25
28	Filtri passa-basso Gaussiani	25
29	Da alto a sinistra senso orario: originale, HPF ideale, HPF di Butterworth e HPF Gaussiano	26
30	Immagine originale (a sinistra) e maschera del filtro Laplaciano (a destra)	27
31	Variazione delle frequenze nell'asse temporale da $t_0 = 0\text{ s}$ a $t_3 = 1\text{ s}$	28
32	Rappresentazione dei problemi della trasformata di Fourier	29
33	Esecuzione della STFT	29
34	Tipologie wavelet fondamentali	30
35	Rappresentazione di una trasformata Wavelet	31
36	Analisi dei dettagli delle serie	32
37	Piramide dell'immagine	33
38	Schema di costruzione della piramide dell'immagine	33
39	Processo di codifica di sottobanda	35
40	Esempio di trasformata Wavelet 2D	36

41	Processo di trasmissione di un'immagine compressa	37
42	Esempio di ridondanza di interpixel	38
43	Esempio di ridondanza psicovisiva	39
44	Esempio di rappresentazione della teoria dell'informazione	39
45	Misura del PSNR in un'immagine	40
46	Schema del modello di compressione di un'immagine	41
47	Esempio tipologia di immagine da comprimere lossless (a sinistra) od anche lossy (a destra)	41
48	Esempio di una codifica di Huffman	42
49	Rappresentazione del bit-plane coding	43
50	Schema del modello di compressione JPEG	44
51	Pattern zig-zag	46
52	Esempio di applicazione ROI dall'immagine originale (a sinistra) all'immagine compressa (a destra)	47
53	Schema del modello di compressione JPEG2000	47
54	Schema della fase di codifica di blocchi	48
55	Esempio di quantizzazione JPEG2000	49
56	Fase di EBCOT	49
57	Strati di qualità	50
58	Processo di adattamento	51
59	Processo di vergenza	51
60	<i>A Rainy Day in Paris</i> che mostra le occlusioni (1), le dimensioni relative (2), le tessiture (3), la prospettiva lineare (4), la prospettiva aerea (5) e le ombre (6)	52
61	Esempio di occlusione	52
62	Esempio di dimensioni relative	53

PARTE I:

IMMAGINI

Capitolo 1:

Elaborazione delle immagini

In questo capitolo viene spiegato come avviene l'elaborazione delle immagini. Per prima cosa, è necessario capire come l'occhio umano cattura e percepisce l'immagine, soprattutto per comprendere quali sono i suoi grandi limiti. Una volta capito ciò, si può procedere all'elaborazione delle immagini.

1.1: Sistema visivo umano

L'occhio umano è racchiuso da tre membrane, dove ognuna ha una funzione rilevante per l'elaborazione delle immagini.

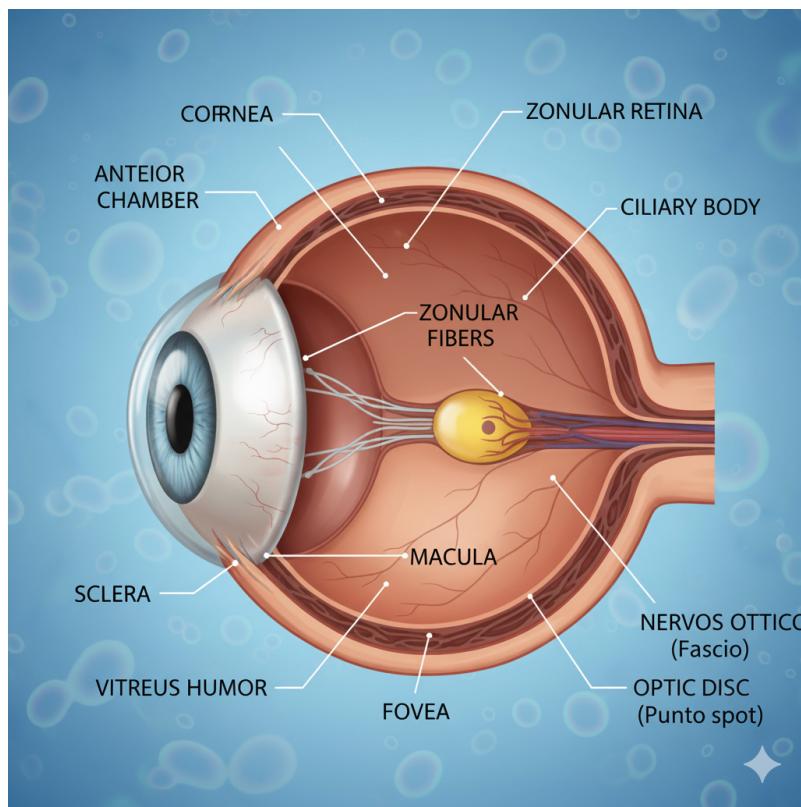


Figura 1: Struttura dell'occhio umano

1.1.1: Cornea e coroide: due membrane come protezione

La cornea è la membrana più esterna dell'occhio umano: infatti, essendo composta da un tessuto resistente e trasparente, è perfetta per racchiudere la superficie anteriore dell'occhio. Inoltre, per ridurre la quantità di luce estranea che entra nell'occhio, è presente la coroide.

1.1.2: Retina: la membrana per la vista

La retina è quella membrana che fornisce il senso della vista all'essere umano. In particolare, permette di mettere a fuoco gli oggetti, grazie alla luce dell'oggetto stesso che entra nella retina.

Inoltre, sono presenti i cosiddetti percettori luminosi, che rendono possibile la visione a pattern.

Il primo percettore sono i coni, che permettono la visione cromatica: infatti possono essere a lunghezza d'onda corta per il blu, a lunghezza d'onda media per il verde ed a lunghezza d'onda lunga per il rosso.

Per la visione acromatica, invece, sono presenti i bastoncelli, per esempio la visione scotopica e la penombra.

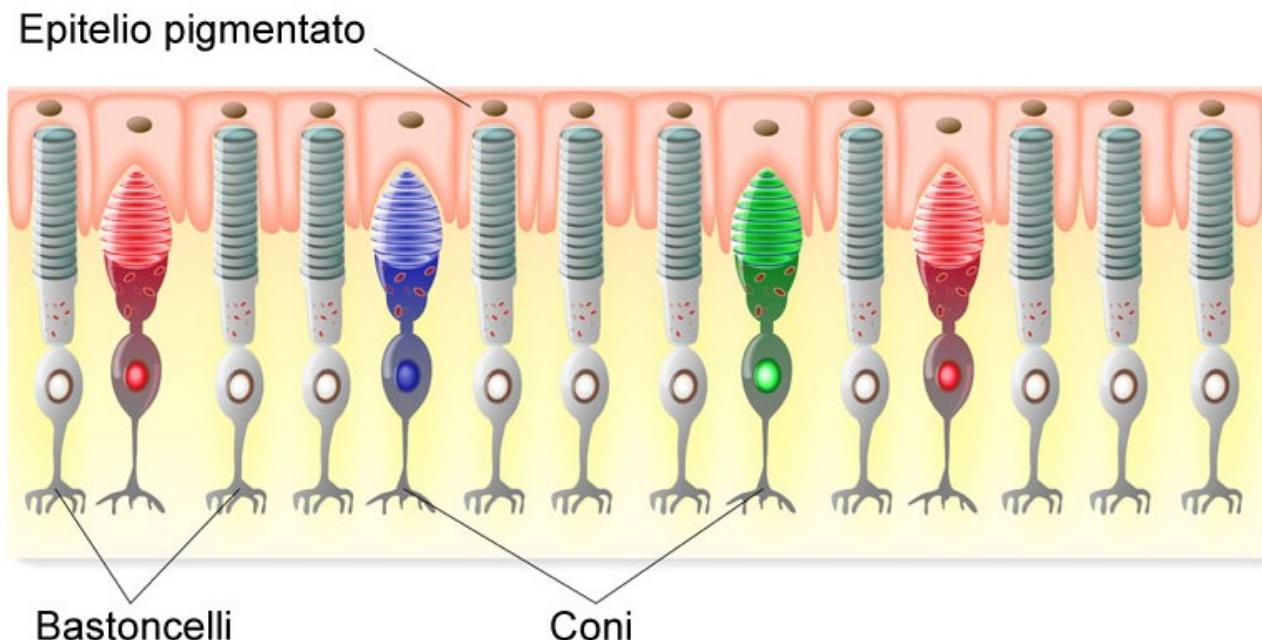


Figura 2: Coni e bastoncelli

1.1.3: Formazione dell'immagine

A questo punto, la formazione dell'immagine avviene nel modo seguente. Le lenti dell'occhio umano sono flessibili: la sua forma è controllata dalle fibre del corpo ciliare.

Inoltre, l'abilità dell'occhio di discriminare i cambiamenti delle intensità di luce a qualsiasi livello di adattibilità specifico, è descritto dalla legge di Weber.

$$k = \frac{\Delta I_C}{I}$$

In particolare:

- k è la costante di Weber, che è un valore costante e caratteristico per ogni specifica modalità sensoriale;
- ΔI_C è la soglia differenziale, ossia la quantità minima di cambiamento affinché il soggetto percepisca una differenza;
- I è l'intensità di riferimento dello stimolo.

Ciò spiega che il sistema visivo tende a sottostimare od a sovrastimare i bordi delle regioni a diverse intensità. Ciò genera le illusioni: esempi sono l'illusione di dolcezza di mais e la griglia di Hermann.

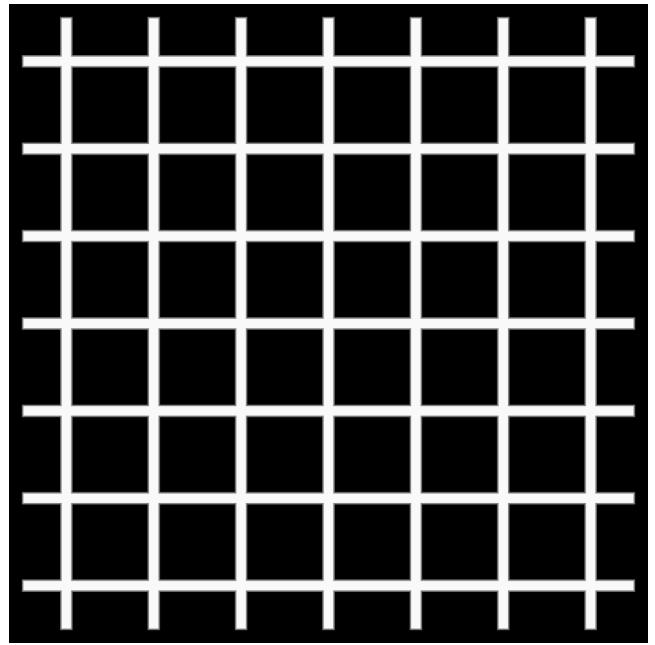


Figura 3: Illusione del mais dolce (a sinistra) e la griglia di Hermann (a destra)

1.2: Rappresentazione delle immagini

Un'immagine è una funzione bidimensionale ($f(x, y)$), definita come il prodotto tra altrettante due funzioni bidimensionali:

- la luminanza ($i(x, y)$), che è la quantità di luce della sorgente incidente sulla scena osservata;
- la riflettanza ($r(x, y)$), che è la quantità di luce riflessa dall'oggetto nella scena.

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

1.2.1: Intensità di un'immagine

L'intensità di un'immagine monocromatica a (x_0, y_0) è il livello di grigio L in quel punto.

$$L = f(x_0, y_0)$$

Il livello di grigio trovato appartiene ad un intervallo dove il minimo corrisponde al nero ed il massimo corrisponde al massimo.

$$L \in [L_{\min}, L_{\max}]$$

1.2.2: Digitalizzazione delle immagini

Per effettuare la digitalizzazione di un'immagine, sono necessari i seguenti parametri:

- M , che è la larghezza dell'immagine;
- N , che è l'altezza dell'immagine;
- L , che è il numero dei livelli di grigi, che è un multiplo di 2^n .

A questo punto, avviene la digitalizzazione, che si compone in due fasi.

La prima fase è il campionamento, che consiste nel suddividere l'immagine in una griglia regolare di punti o celle. Ogni cella produce un pixel, che è l'unità minima di un'immagine digitale. A questo punto viene misurata la risoluzione in dpi, che non è altro che la densità della griglia: più è fitta più l'immagine sarà fedele all'originale.

La seconda ed ultima fase è la quantizzazione, che è il processo di discretizzazione dell'intensità. In particolare, ad ogni pixel gli viene assegnato un valore numerico discreto che ne codifica il livello di grigio. Infine, il valore numerico viene convertito in una stringa binaria.

Il risultato della digitalizzazione, produce una matrice di numeri reali di dimensioni $M \times N$.

$$f(x, y) = \begin{bmatrix} f(0, 0) & \dots & f(0, N - 1) \\ \vdots & \ddots & \vdots \\ f(M - 1, 0) & \dots & f(M - 1, N - 1) \end{bmatrix}$$

1.3: Manipolazione delle immagini

La manipolazione delle immagini si pone come obiettivo principale quello di modificare l'immagine originale che si adatta di più al contesto richiesto: come sfocare lo sfondo in un ritratto oppure rendere più luminose le vene in un'immagine catturata da un'esame medico.

In questo paragrafo, vengono elencati una serie di possibili manipolazioni, andando a modificare la $f(x, y)$ con un operatore T definito con i suoi vicini (x, y) .

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

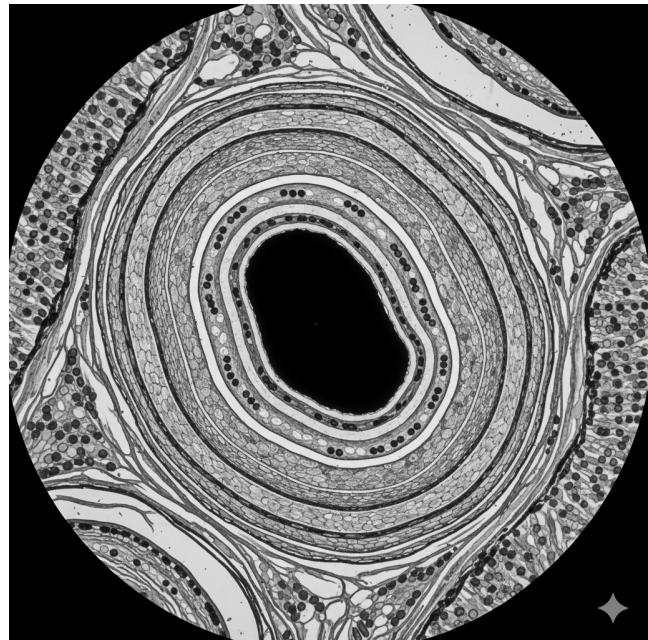


Figura 4: Esempi di possibili manipolazioni di immagini

In particolare, si considera:

- r è il livello di grigio di input;
- s è il livello di griglio di output.

1.3.1: Manipolazioni principali

Di seguito ne è riportato un grafico che ne riporta le principali tipologie di manipolazione delle immagini.

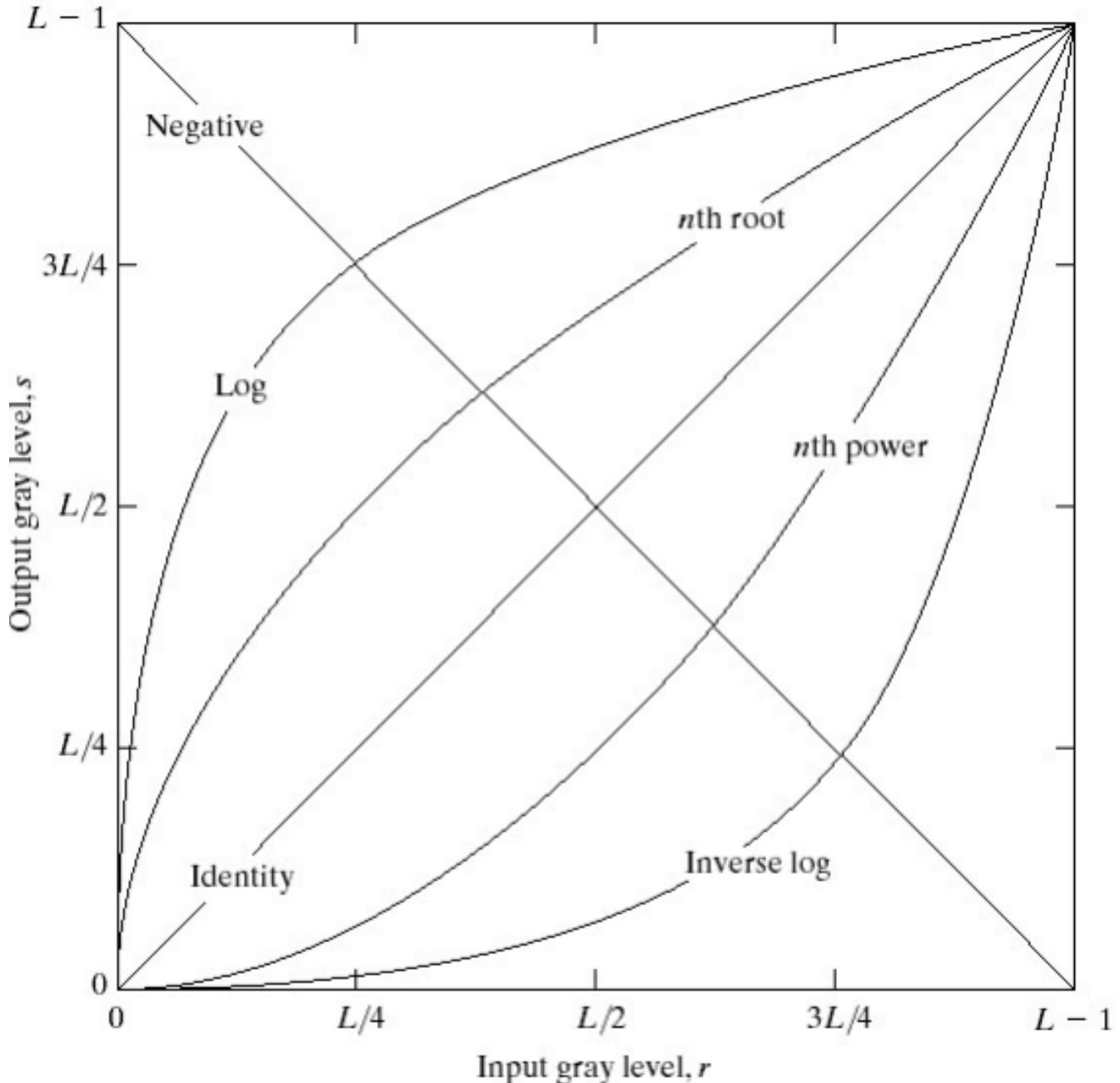


Figura 5: Possibili manipolazioni delle immagini

La prima ed anche la più semplice è la funzione identità, in cui il livello di grigio in entrata corrisponde a quello d'uscita, perciò l'immagine non riporta alcuna alterazione. Tale funzione è una retta che va da $(0, 0)$ a $(L - 1, L - 1)$.

$$s = r$$

La seconda è la funzione negativa, in cui inverte i livelli di grigio, creando la cosiddetta immagine negativa. Questa funzione non è altro che una retta che va da $(0, L - 1)$ a $(L - 1, 0)$.

$$s = L - 1 - r$$

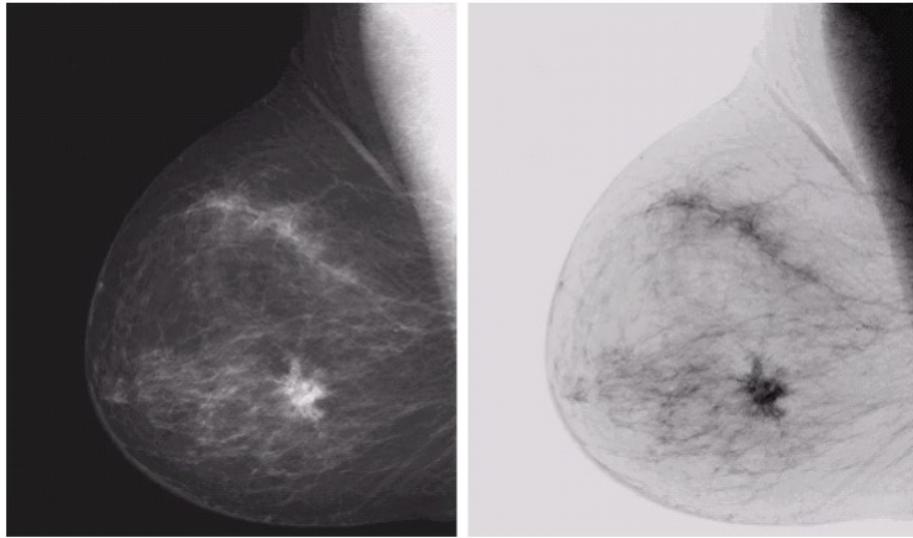


Figura 6: Immagine originale (a sinistra) e immagine in negativo (a destra)

Tale funzione risulta molto utile per risaltare regioni di grigio incorporate da regioni scure.

La terza funzione è la funzione logaritmica, che tende ad espandere i valori dei pixel scuri ed ad comprimere i pixel chiari. Se invece si vuole ottenere l'opposto, allora si usa la funzione logaritmica inversa. La prima equazione è la funzione logaritmica, mentre la seconda è la funzione logaritmica inversa. Entrambi sfruttano la costante di scala $r \in [0, L - 1]$.

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

$$s = \exp\left(\frac{r}{c}\right) - 1$$

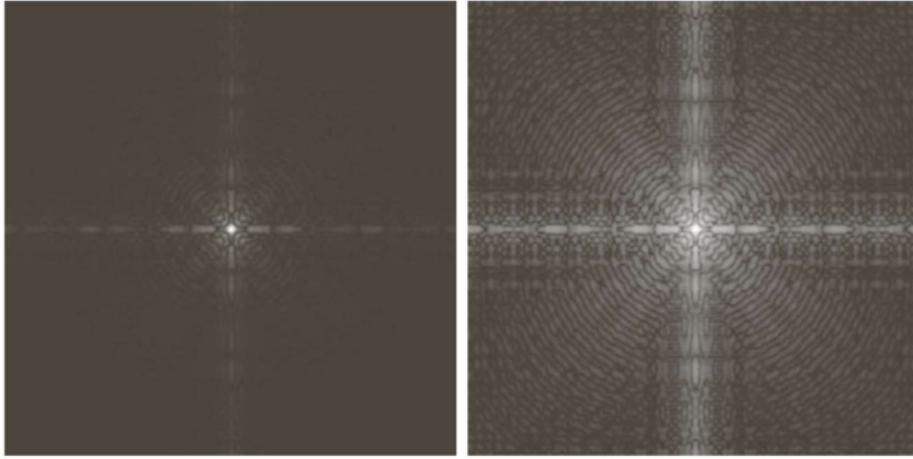


Figura 7: Esempio di due immagini a cui sono state applicate con r differenti

L'ultimo gruppo di funzioni riguarda la correzione gamma, che è un'operazione non lineare usata per codificare e decodificare i valori di luminanza in un sistema di visualizzazione di immagini. Esse sfruttano le costanti c e γ , entrambe positive.

$$s = cr^\gamma$$

Ciò avviene per due motivi principali:

- l'occhio umano non percepisce la luminosità in modo lineare, perciò le immagini risulterebbero scure e con pochi dettagli nelle ombre;
- gli schermi più datati, come quelli a tubo catodico, hanno una risposta non lineare alla tensione in ingresso, seguendo una legge di potenza, e si è scelto di mantenere ciò anche per gli schermi più moderni per mantenere compatibilità e coerenza visiva per la percezione umana.

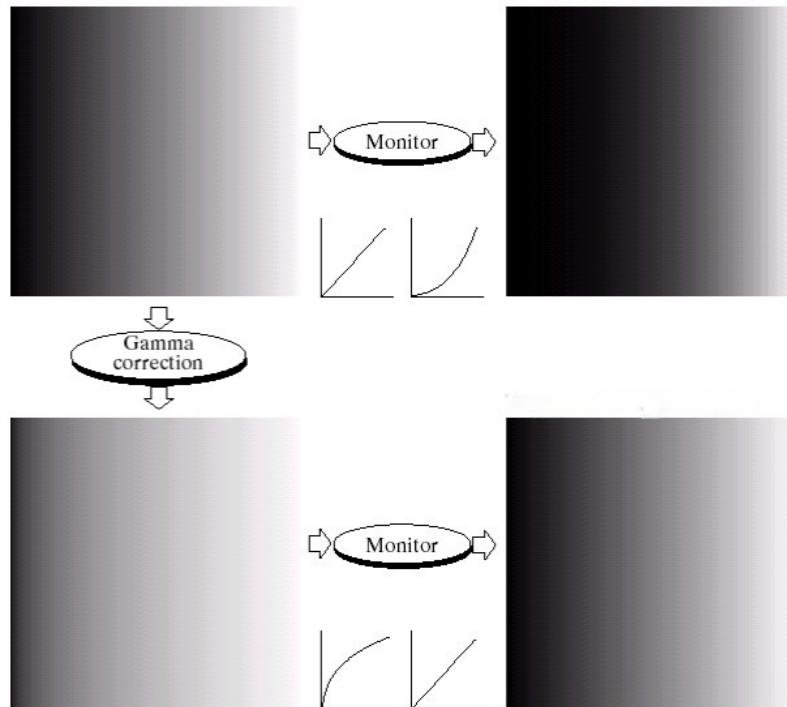


Figura 8: Realizzazione della correzione gamma sui monitor

Di seguito, viene riportata una figura che dimostra quando sia importante la calibrazione corretta della correzione gamma sui monitor.

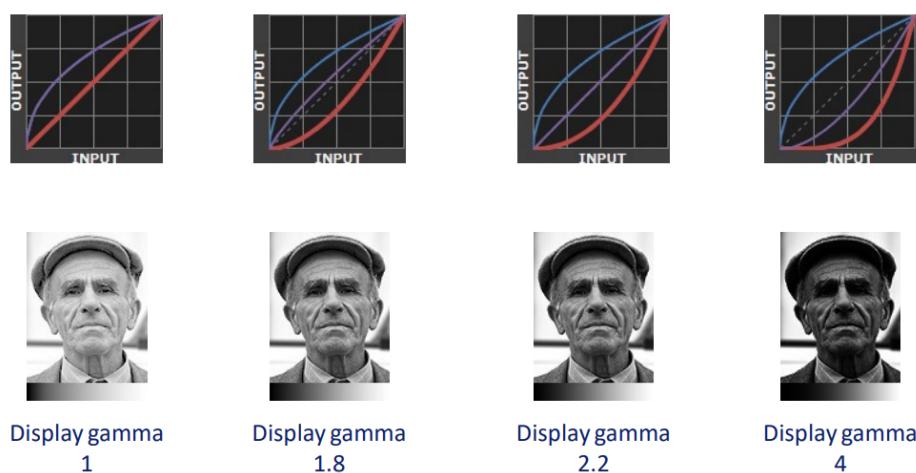


Figura 9: Calibrazione del gamma (azzurro la gamma dell'immagine, viola la gamma del display e la rossa la gamma complessiva)

Nella prima colonna non è stata applicata alcuna correzione; nella seconda l'applicazione è insufficiente, generando un'immagine troppo chiara; nella terza è l'applicazione ideale; infine nell'ultima l'applicazione è eccessiva generando un'immagine troppo scura.

1.3.2: Funzione di trasformazione lineare a tratti

La funzione di trasformazione lineare a tratti ha la particolarità di non essere descritta da una singola equazione per tutto l'intervallo dei livelli di grigio, ma da più segmenti lineari, ognuno applicabile ad un intervallo di livello di grigio ben specifico. Il vantaggio sta nell'avere una forma con livello di complessità a scelta, a discapito, però, nell'avere più input dall'utente.

La prima funzione di trasformazione lineare a tratti è il contrast stretching, che si pone come obiettivo quello di aumentare il contrasto di un'immagine che appare sbiadita o con scarso intervallo dinamico. A questo punto, si scelgono due punti (r_1, s_1) e (r_2, s_2) in cui i livelli sotto r_1 (molto scuri) oppure sopra r_2 (molto chiari) vengono compressi, in altre parole vengono mappati rispettivamente quasi a 0 e a $L - 1$. Mentre se è compreso tra r_1 ed r_2 viene aumentato il contrasto, espandendo i livelli di grigio sull'intero intervallo di uscita.

$$s = \begin{cases} \frac{s_1}{r_1} r & \text{se } 0 \leq r < r_1 \\ \frac{s_2 - s_1}{r_2 - r_1}(r - r_1) + s_1 & \text{se } r_1 \leq r \leq r_2 \\ \frac{L - 1 - s_2}{L - 1 - r_2}(r - r_1) + s_2 & \text{se } r_2 < r \leq L - 1 \end{cases}$$

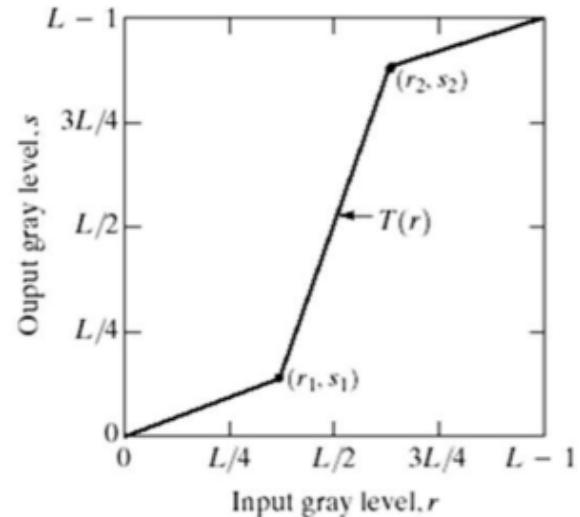


Figura 10: Definizione analitica e grafico del contrast stretching

Un caso limite si ha se avviene un'immagine binaria (bianco o nero), avendo il cosiddetto thresholding.

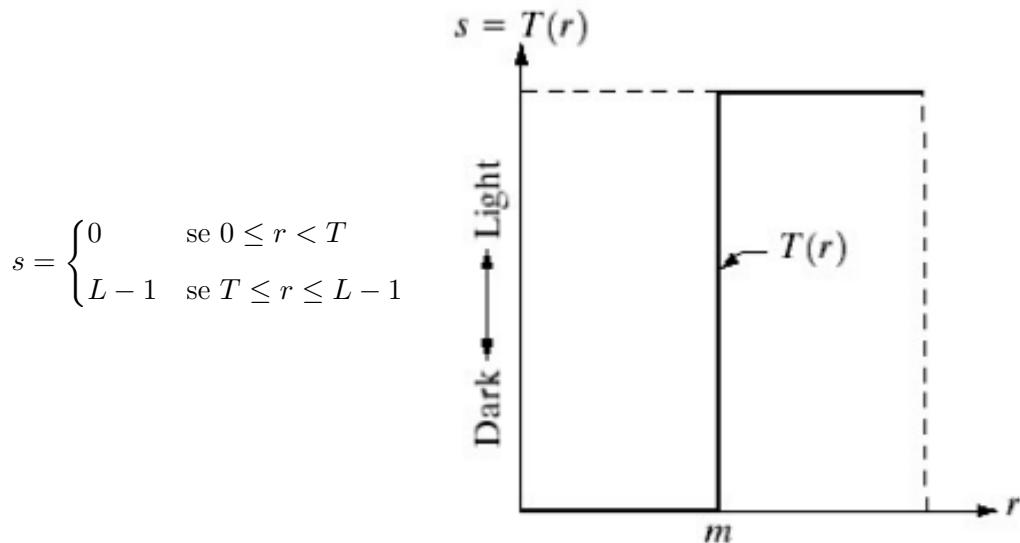


Figura 11: Definizione analitica e grafico del thresholding

La seconda funzione di trasformazione lineare a tratti è il gray-level slicing, che si pone come obiettivo quello di evidenziare un intervallo di livello di grigio ben specifico e sopprimere tutti gli altri. In particolare si fa riferimento a due casi ben specifici:

- sfondo costante, in cui l'intervallo viene mappato ad un livello alto ($L - 1$) ed il resto al livello 0;

$$s = s_{high} \text{rect}_{B-A} \left(r - \frac{A+B}{2} \right)$$

- sfondo invariato, in cui l'intervallo viene mappato ad un livello alto ($L - 1$) ed il resto viene lasciato invariato.

$$s = r + (s_{high} - r) \text{rect}_{B-A} \left(r - \frac{A+B}{2} \right)$$

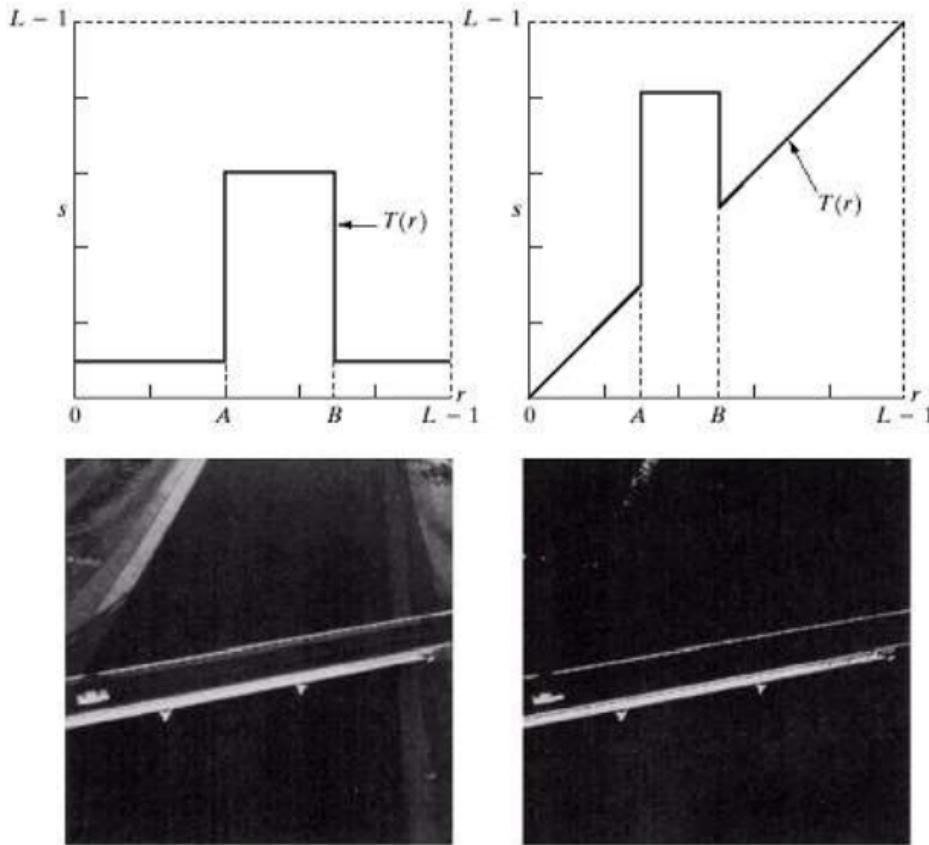


Figura 12: Gray-level slicing a sfondo costante (a sinistra) e a sfondo invariato (a destra)

La terza ed ultima funzione di trasformazione lineare a tratti è il bit-plain slicing, che consiste nel mettere evidenza solamente i bit più significativi (che sono i primi bit).

1.3.3: Equalizzazione dell'istogramma

Quando si parla di un'immagine, è possibile costruire l'istogramma dell'immagine stessa, che non è altro che il numero di pixel che contengono un determinato valore di livello di grigio per ogni livello n_k .

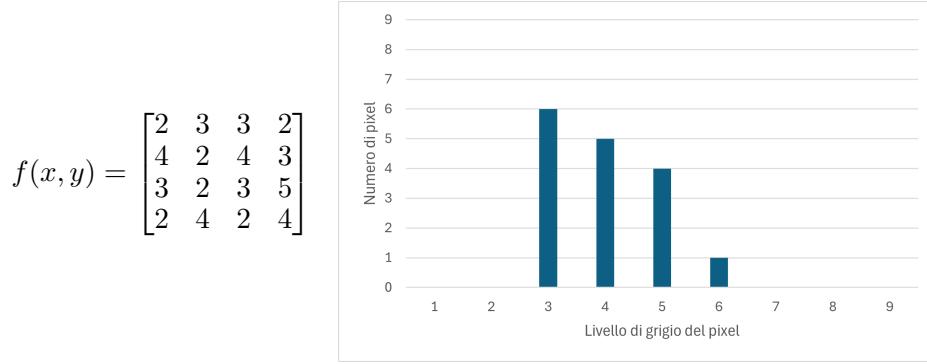


Figura 13: Esempio di istogramma di un’immagine 4x4 con livello di grigio [0, 9]

In termini statistici, è preferibile rappresentare l’istogramma normalizzato, ossia dividere n_k con il numero totale di pixel n .

$$p(n_k) = \frac{n_k}{n}$$

A questo punto, la tecnica dell’equalizzazione dell’istogramma consiste nel cambiare l’istogramma dell’immagine in un’istogramma uniforme, in cui la percentuale di ogni livello di grigio rimane sempre la stessa. Per effettuare ciò, occorre dei fondamenti di probabilità e statistica e di calcolo integrale.

Per prima cosa, questa operazione effettua una trasformata, perciò si può scrivere ciò come $s = T(r)$ e siccome la trasformata è reversibile, esiste anche la sua inversa $r = T^{-1}(s)$. Per semplicità, risulta molto conveniente studiare ciò nel continuo. Dati rispettivamente, $p_{in}(r)$ e $p_{out}(s)$, come la probabilità di livello di grigi di input e di output, dalla teoria di probabilità, si ha la formula seguente (per $0 \leq r \leq L - 1$ e $0 \leq s \leq L - 1$).

$$p_{out}(s) = \left[p_{in}(r) \frac{ds}{dr} \right]_{r=T^{-1}(s)}$$

A questo punto, la trasformata di $p_{in}(r)$ è pari alla formula seguente:

$$s = T(r) = \int_0^r p_{in}(r') dr', 0 \leq r \leq 1$$

che è la funzione di distribuzione cumulativa (CDF). Perciò, dal teorema fondamentale del calcolo, si sfrutta la seguente formula.

$$p_{in}(r) = \frac{dr}{ds}$$

Infine, facendo dei semplici conti e sfruttando le proprietà delle derivate delle funzioni inverse, si ricava che:

$$p_{out}(s) = \left[p_{in}(r) \frac{ds}{dr} \right]_{r=T^{-1}(s)} = \left[p_{in}(r) \frac{1}{\frac{dr}{ds}} \right]_{r=T^{-1}(s)} = \left[p_{in}(r) \frac{1}{p_{in}(r)} \right]_{r=T^{-1}(s)} = [1]_{r=T^{-1}(s)} = 1, 0 \leq s \leq 1$$

la densità di probabilità in uscita risulta uniforme.

Quindi per effettuare l’equalizzazione dell’istogramma:

1. per ogni pixel si calcola il $p_{in}(r_k)$;

$$p_{in}(r_k) = \frac{n_k}{n}, 0 \leq r_k \leq 1, 0 \leq k \leq L - 1$$

2. basandosi sulla CDF, si esegue la trasformata discreta.

$$s_k = T_{r_k} = \sum_{j=0}^k p_{in}(r_j), 0 \leq k \leq L - 1$$

Tornando all'esempio precedente, si effettua la CDF nel modo seguente.

$$f(x, y) = \begin{bmatrix} 2 & 3 & 3 & 2 \\ 4 & 2 & 4 & 3 \\ 3 & 2 & 3 & 5 \\ 2 & 4 & 2 & 4 \end{bmatrix}$$

k	r_k	n_k	$p_{in}(r_k)$
2	0	6	3/8
3	1/3	5	5/16
4	2/3	4	1/4
5	1	1	1/16

Figura 14: Calcolo dell'istogramma normalizzato

$$\begin{aligned} s_2 &= p_{in}(r_2) = \frac{3}{8} \rightarrow 0 \\ s_3 &= p_{in}(r_3) = \frac{3}{8} + \frac{5}{16} = \frac{11}{16} \rightarrow \frac{2}{3} \\ s_4 &= p_{in}(r_4) = \frac{3}{8} + \frac{5}{16} + \frac{1}{4} = \frac{15}{16} \rightarrow 1 \\ s_5 &= p_{in}(r_5) = \frac{3}{8} + \frac{5}{16} + \frac{1}{4} + \frac{1}{16} = 1 \rightarrow 1 \end{aligned}$$

Perciò:

- al livello 2 si associa il livello 2;
- al livello 3 si associa il livello 4;
- al livello 4 si associa il livello 5;
- al livello 5 si associa il livello 5.

A questo punto, si ottiene l'immagine equalizzata, come mostrato di seguito.

$$g(x, y) = \begin{bmatrix} 2 & 4 & 4 & 2 \\ 5 & 2 & 5 & 4 \\ 4 & 2 & 4 & 5 \\ 2 & 5 & 2 & 5 \end{bmatrix}$$

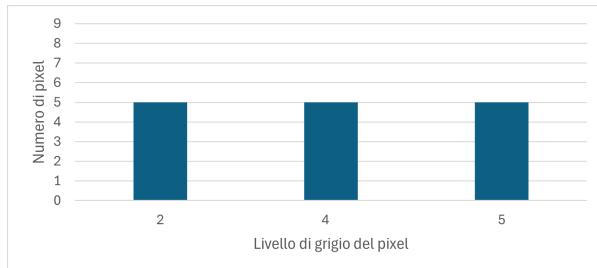


Figura 15: Esempio di istogramma di un'immagine equalizzata

Capitolo 2:

Filtri nel dominio spaziale

Nelle telecomunicazioni, per filtrare un segnale nel dominio del tempo, si l'operazione di convoluzione.

$$x(t) * h(t) = \int_{-\infty}^{\infty} h(\tau)x(t - \tau) d\tau \quad x[n] * h[n] = \sum_{i=-\infty}^{\infty} h[i]x[n-i]$$

Per quanto riguarda le immagini, nel dominio spaziale, la convoluzione avviene in due dimensioni (somma del prodotto elemento per elemento delle due matrici), dove:

- l'immagine è il segnale d'ingresso;
- il filtro è il nucleo della convoluzione, detto maschera.

$$g(x, y) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} h(i, j)f(x - i, y - j)$$

Per esempio, data l'immagine e il filtro seguente:

$$f(x, y) = \begin{bmatrix} 5 & 8 & 3 & 4 \\ 3 & 2 & 1 & 1 \\ 0 & 9 & 5 & 3 \\ 4 & 2 & 7 & 2 \end{bmatrix} \quad h(x, y) = \begin{bmatrix} 2 & 1 & 0 \\ 1 & 1 & -1 \\ 0 & -1 & -2 \end{bmatrix} \quad h_{flip}(x, y) = - \begin{bmatrix} 2 & 1 & 0 \\ 1 & 1 & -1 \\ 0 & -1 & -2 \end{bmatrix} = \begin{bmatrix} -2 & -1 & 0 \\ -1 & -1 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$

si mette all'inizio in posizione $(0, 0)$ dell'immagine e quello è il punto centrale dell'immagine. Come si può notare, sono presenti dei punti dell'immagine che vanno fuori dall'immagine. Per evitare ciò, è possibile:

- ignorare i bordi, partendo dai punti in cui si ha una piena sovrapposizione dell'immagine;
- si assumono i valori fuori dai bordi pari a 0.

$$g(0, 0) = \begin{bmatrix} -2 & -1 & 0 \\ -1 & -1 & 1 \\ 0 & 1 & 2 \end{bmatrix} * \begin{bmatrix} 0 & 0 & 0 \\ 0 & 5 & 8 \\ 0 & 3 & 5 \end{bmatrix} = -1 \times 5 + 1 \times 8 + 1 \times 3 + 5 \times 5 = 20$$

L'immagine finale, ponendo i punti fuori dall'immagine pari a 0, è dunque la seguente.

$$g(x, y) = \begin{bmatrix} 20 & 10 & 2 & 2 \\ 18 & 1 & -8 & -7 \\ 14 & 22 & 5 & -3 \\ 6 & -4 & -16 & -18 \end{bmatrix}$$

Nel caso in cui si ignorassero i bordi, si applicherebbe il filtro solamente nei punti $(1, 1)$, $(1, 2)$, $(2, 1)$ e $(2, 2)$. Come si può intuire, l'immagine di output viene tagliata.

$$g(x, y) = \begin{bmatrix} 3 & 10 \\ -4 & 5 \end{bmatrix}$$

Infine, eseguire convoluzioni con maschere risulta essere molto versatile, poiché a seconda dei coefficienti della maschera, si possono ottenere risultati differenti: per esempio la sfocatura, il contrasto od il rilevamento dei bordi. In questo capitolo, vengono spiegati i filtri di smussamento e di nitidezza.

2.1: Filtri di smussamento

I filtri di smussamento permettono di gestire la sfocatura dell'immagine e della riduzione del rumore. In particolare, la sfocatura è un processo che serve per rimuovere piccoli dettagli e colmare piccoli scalini in linee e curve. La sfocatura accompagna la riduzione del rumore.

2.1.1: Filtri di smussamento lineari

I filtri di smussamento lineari applicano la media dei pixel del vicinato. Essi ripiazzano il valore di ogni pixel con la media dei livello di grigio definiti dalla maschera. In particolare, a seconda dei valori della maschera si hanno tre tipologie di filtro.

La prima tipologia è il filtro media aritmetica, in cui la maschera contiene il prodotto scalare tra l'inverso del numero di elementi della maschera e la maschera stessa di tutti 1.

$$h(x, y) = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad h(x, y) = \frac{1}{25} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Tale filtro applica una sfocatura uniforme all'immagine: più la maschera contiene elementi, maggiore sarà la sfocatura. La seconda tipologia è il filtro media ponderata, in cui si vuole dare più importanza a dei pixel rispetto ad altri, ad esempio il pixel centrale.

$$h(x, y) = \begin{bmatrix} 3/40 & 1/8 & 3/40 \\ 1/8 & 1/5 & 1/8 \\ 3/40 & 1/8 & 3/40 \end{bmatrix}$$

La terza ed ultima tipologia è il filtro gaussiano, in cui la maschera è una gaussiana a due dimensioni, in cui la deviazione standard (σ) ne determina la larghezza della campana.

$$h(x, y) = \exp \left[\frac{-(x^2 + y^2)}{2\sigma^2} \right]$$

La deviazione standard controlla l'intensità della sfocatura: più è alto più sfoca. Comunque, essa sfoca molto meno brutalmente rispetto al filtro media.

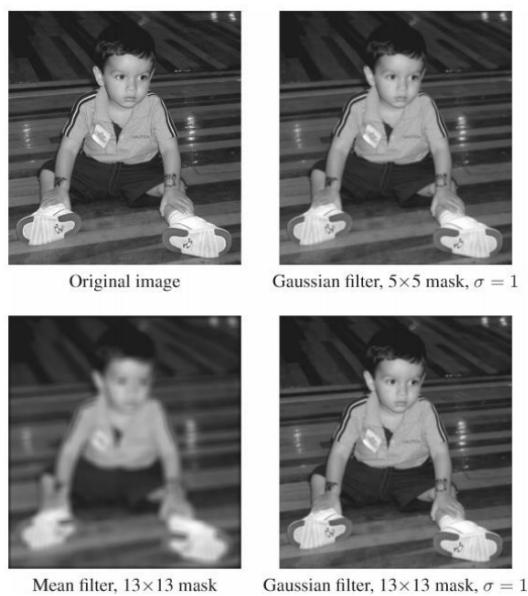


Figura 16: Filtro media e gaussiano

2.1.2: Filtri di smussamento non lineari

I filtri di smussamento non lineari non presentano un'operazione di convoluzione vera e propria, ma si basano su ordinamento statistico: in particolare, in base al ranking dei pixel. I pixel che sono considerati non rappresentativi vengono eliminati. Inoltre, vengono sostituiti i valori dei pixel con il valore della classifica. Questa tipologia viene usata per la riduzione del rumore, sfocando però l'immagine.

Il filtro di smussamento non lineare più comune è il filtro mediano, che consiste nel:

1. prendere una porzione di un'immagine (solitamente 3x3 o 5x5);

$$\begin{bmatrix} 9 & 12 & 0 \\ 5 & 5 & 9 \\ 8 & 10 & 7 \end{bmatrix}$$

2. convertire tale porzione in un array ordinato;

$$\{0, 5, 5, 7, 8, 9, 9, 10, 12\} \rightarrow 8$$

3. sostituire la mediana dell'array nella porzione presa, sfocando l'immagine;

$$\begin{bmatrix} 8 & 8 & 8 \\ 8 & 8 & 8 \\ 8 & 8 & 8 \end{bmatrix}$$

4. iterare i passaggi precedenti per tutta l'immagine.

Questo filtro permette di rimuovere il rumore sale e pepe, molto comune nelle foto più datate, in cui l'immagine presenta dei pixel completamente bianchi e neri: infatti, tali pixel non saranno mai dei mediani: per questo motivo vengono eliminati. Inoltre, la sfocatura non è così evidente, un altro lato positivo da non sottovalutare.



Figura 17: Rimozione del rumore sale e pepe con un filtro mediano (sinistra originale e destra modificata)

2.2: Filtri di nitidezza

I filtri di shapening preservano i dettagli, andando ad evidenziare i bordi. Per fare ciò, è necessario trovare quella operazione che permetta di distinguere quali sono i pixel dell'immagine che sono uguali oppure molto simili. Tale operazioni sono:

- la derivata prima, che esegue la differenza tra il pixel successivo ed il pixel corrente;

$$\frac{\delta f}{\delta x} = f(x+1) - f(x)$$

- la derivata seconda, che esegue la somma tra il pixel successivo ed il pixel precedente e infine sottrae due volte il pixel corrente.

$$\frac{\delta^2 f}{\delta x^2} = f(x+1) + f(x-1) - 2f(x)$$

Per tali scopi, risulta molto conveniente usare la derivata seconda, poiché presenta una risposta più forte per i dettagli (facilmente intuibile) ed ha un'implementazione decisamente più semplice (poco intuibile).

2.2.1: Il Laplaciano

Il filtro di nitidezza più usato è il Laplaciano, che sfrutta il gradiente secondo dell'immagine: ciò è decisamente logico, dato che in più di una dimensione e con le derivate parziale, si fa riferimento ad ogni derivata parziale, perciò è necessario il gradiente.

$$\nabla^2 f = \frac{\delta^2 f}{\delta x^2} + \frac{\delta^2 f}{\delta y^2}$$

Con dei calcoli matematici si ricava la formula seguente:

$$\frac{\delta^2 f}{\delta x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

$$\frac{\delta^2 f}{\delta y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

$$\nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

da cui si ricava la maschera, ricavando i coefficienti del gradiente.

$$\nabla^2 f = \begin{bmatrix} f(x-1, y-1) & f(x-1, y) & f(x-1, y+1) \\ f(x, y-1) & f(x, y) & f(x, y+1) \\ f(x+1, y-1) & f(x+1, y) & f(x+1, y+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Tuttavia, tale filtro trova solamente i bordi dell'immagine. Per questo motivo, l'immagine di output è solamente uno dei fattori della somma algebrica con l'immagine originale.

$$g(x, y) = f(x, y) + c\nabla^2 [(x, y)]$$

In particolare, $c = 1$ se il coefficiente centrale è positivo; $c = -1$ altrimenti. A questo punto, la nuova immagine è la seguente.

$$g(x, y) = f(x, y) - f(x+1, y) - f(x-1, y) - f(x, y+1) - f(x, y-1) + 4f(x, y)$$

$$g(x, y) = 5f(x, y) - f(x+1, y) - f(x-1, y) - f(x, y+1) - f(x, y-1)$$

Perciò, la maschera è la seguente.

$$h(x, y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Infatti, ora si comprende il motivo per cui la sua semplicità di implementazione. Inoltre, il Laplaciano è presentato i seguenti vantaggi: isotopico: perciò è indipendente dal punto in cui viene applicato, poiché applica gli array circolare, quindi la rotazione resta invariata.

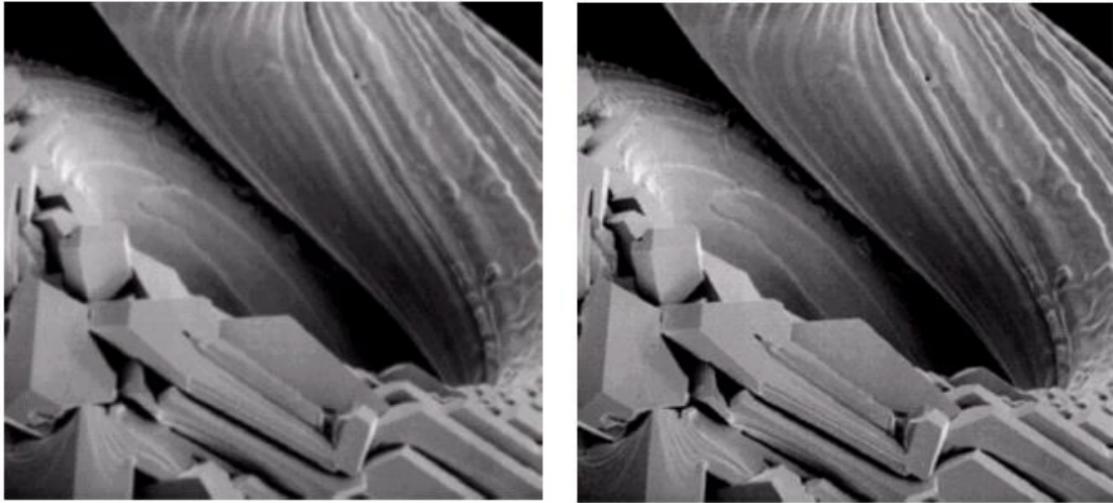


Figura 18: Applicazione del Laplaciano (sinistra originale e destra modificata)

2.2.2: Filtri di contrasto

Nel caso serve capire una scala di dettagli da evidenziare all'immagine, piuttosto che sfruttare un'operazione fissa, conviene usare i filtri di contrasto (in inglese unsharp mask). Ciò consiste in:

1. si applica un filtro Gaussiano;

$$\bar{f}(x, y) = f(x, y) * \exp \left[\frac{-(x^2 + y^2)}{2\sigma^2} \right]$$

2. si sottrae l'immagine originale con l'immagine filtrata con il filtro gaussiano, ottenendo la maschera dell'immagine;

$$g_{mask}(x, y) = f(x, y) - \bar{f}(x, y)$$

3. si aggiunge all'immagine originale la maschera dell'immagine moltiplicata per un fattore di scala di dettagli k .

$$g(x, y) = f(x, y) + kg_{mask}(x, y)$$

Ciò si dimostra nel modo seguente.

$$g(x, y) = f(x, y) + kg_{mask}(x, y) = f(x, y) + k[f(x, y) - \bar{f}(x, y)] = (1 + k)f(x, y) - \bar{f}(x, y)$$

Siccome l'immagine mascherata è un'ottima approssimazione del Laplaciano ed il filtro guassiano si può approssimare con un filtro medio 3x3, si approssima $c \approx 1+k$, che prende il nome di fattore di amplificazione.

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & c & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Ciò prende il nome di high-boost filtering.

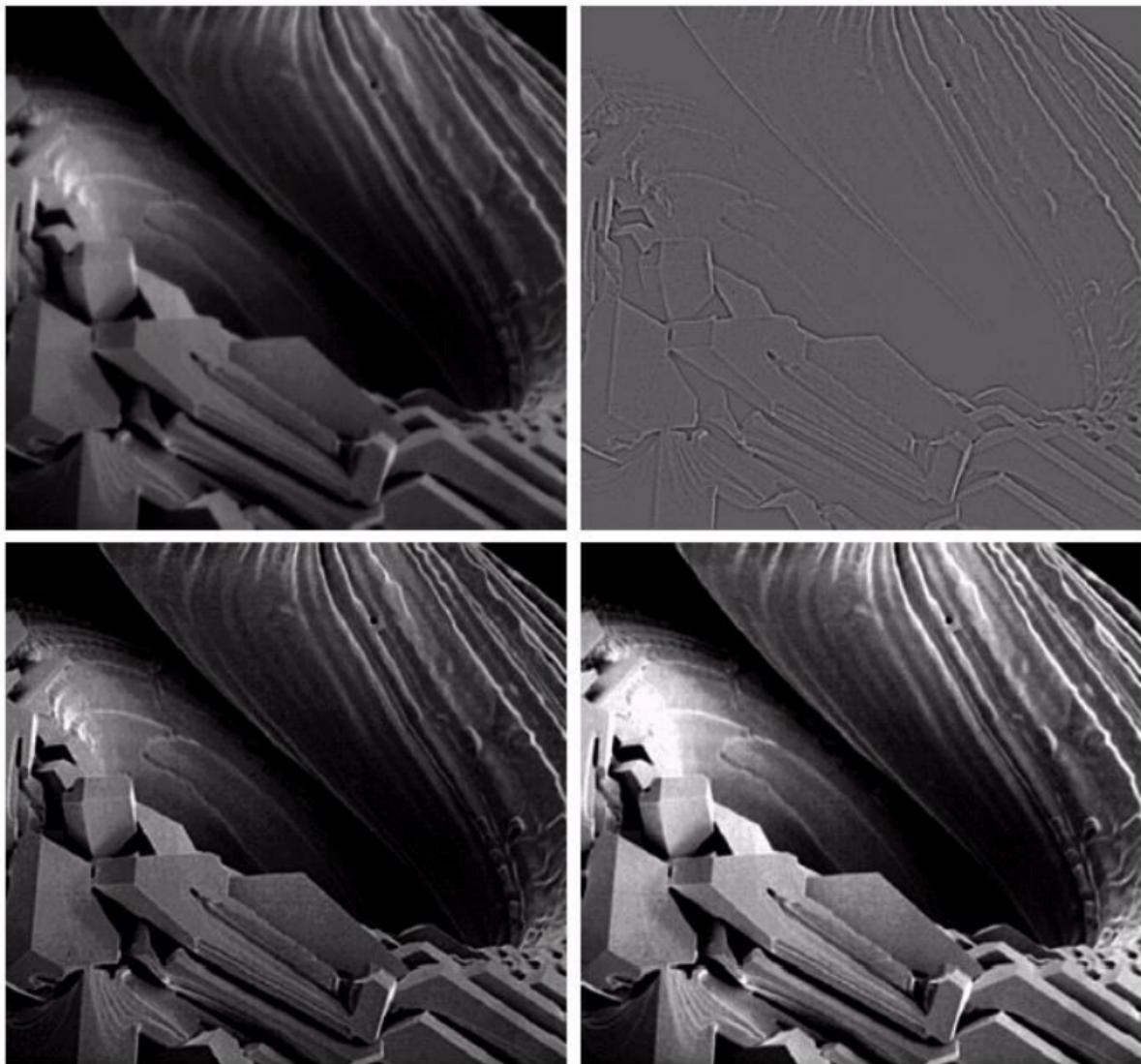


Figura 19: Originale, Laplaciano filtro di contrasto e high-boost filtering da in alto a sinistra in senso orario

2.2.3: Filtri di rilevamento dei bordi

Per rilevare i bordi, sfruttare la derivata seconda è molto semplice da implementare. Tuttavia, presenta un problema da non sottovalutare: è molto sensibile al rumore. Per questo motivo, se l'obiettivo è quello di rilevare i bordi, conviene invece usare la derivata prima. Inoltre, in matematica, la derivata prima rileva i picchi: per questo motivo, è l'operazione che permette di rilevare i bordi. Tuttavia, va eseguito sia per le righe che per le colonne.

Il primo nucleo prende il nome di operatori di Roberts, in cui si approssima attraverso l'equivalenza digitale della derivata del primo ordine, prendendo in considerazione la differenza tra il pixel successivo e il pixel precedente lungo la riga e lungo la colonna.

$$h_x(x, y) \approx f(x + 1, y) - f(x - 1, y)$$

$$h_y(x, y) \approx f(x, y + 1) - f(x, y - 1)$$

Ciò si può rappresentare anche con due matrici 2x2.

$$h_x = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

$$h_y = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

Di seguito, è riportato un esempio con l'immagine 3x3 seguente.

$$f(x, y) = \begin{bmatrix} 10 & 20 & 30 \\ 40 & 50 & 60 \\ 70 & 80 & 90 \end{bmatrix}$$

Adesso, si applicano le due convoluzioni: viene mostrato solo un prodotto per riga e per colonna.

$$h_x(1, 1) = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} * \begin{bmatrix} 10 & 20 \\ 40 & 50 \end{bmatrix} = 0 \times 10 + (-1) \times 20 + 1 \times 40 + 0 \times 50 = 20$$

$$h_y(1, 1) = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} * \begin{bmatrix} 10 & 20 \\ 40 & 50 \end{bmatrix} = -1 \times 10 + 0 \times 20 + 0 \times 40 + 1 \times 50 = 40$$

Tuttavia, si preferisce usare la coppia di kernel di Prewitt e di Sobel, riportati di seguito rispettivamente.

Prewitt

$$h_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad h_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Sobel

$$h_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad h_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Tali nuclei condividono le seguenti due proprietà:

- hanno i coefficienti opposti per avere un'alta risposta nella regione dell'immagine con molta variazione d'intensità, in cui la probabilità che ci sia un bordo è molto alta;
- la somma dei coefficienti è pari a 0: ciò significa che quando è applicato ad un'immagine con una regione completamente omogenea, il risultato è 0.

Una volta fatto ciò, si calcolano:

- la magnitudine, che misura quanto è forte il bordo;

$$h = \sqrt{h_x^2 + h_y^2}$$

- la direzione, che misura l'orientamento del bordo.

$$\theta = \text{atan2}(h_y, h_x) = \begin{cases} \arctan\left(\frac{h_y}{h_x}\right) & \text{se } h_x > 0 \\ \arctan\left(\frac{h_y}{h_x}\right) + \pi & \text{se } h_x < 0 \wedge h_y \geq 0 \\ \arctan\left(\frac{h_y}{h_x}\right) - \pi & \text{se } h_x < 0 \wedge h_y < 0 \\ \frac{\pi}{2} & \text{se } h_x = 0 \wedge h_y > 0 \\ -\frac{\pi}{2} & \text{se } h_x = 0 \wedge h_y < 0 \\ 0 & \text{se } h_x = 0 \wedge h_y = 0 \end{cases}$$

Combinando questi due risultati, è possibile comprendere bene la forza effettiva del bordo.

A seconda del kernel utilizzato, è possibile applicare dei filtri di rilevamento dei bordi. In questo corso ne vengono analizzati di due tipologie.

Il primo, che è anche quello più potente disponibile ad oggi, è il rilevatore di bordi di Canny. L'algoritmo consiste nei seguenti passaggi:

1. si applica un filtro gaussiano per ridurre il rumore;

$$\bar{f}(x, y) = f(x, y) * \exp \left[\frac{-(x^2 + y^2)}{2\sigma^2} \right]$$

2. si calcolano i gradienti locali con la coppia di kernel di Sobel;

$$h_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$h_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$h = \sqrt{h_x^2 + h_y^2}$$

$$\theta = \text{atan2}(h_y, h_x)$$

3. avviene la cosiddetta soppressione dei non-massimi, in cui si guarda la direzione e si confronta la magnitudine calcolata con i suoi vicini, che viene mantenuta solamente se è un massimo locale;
4. si effettua un'ulteriore processo decisionale a soglia, per classificare un pixel come bordo forte ($h(p) > T_{alta}$), bordo debole ($h(p) < T_{bassa}$) oppure un non bordo ($T_{bassa} \leq h(p) \leq T_{alta}$);
5. si prendono solo i bordi forti oppure i bordi deboli adiacenti (anche in diagonale) con un bordo forte, il resto dei bordi deboli vengono scartati.

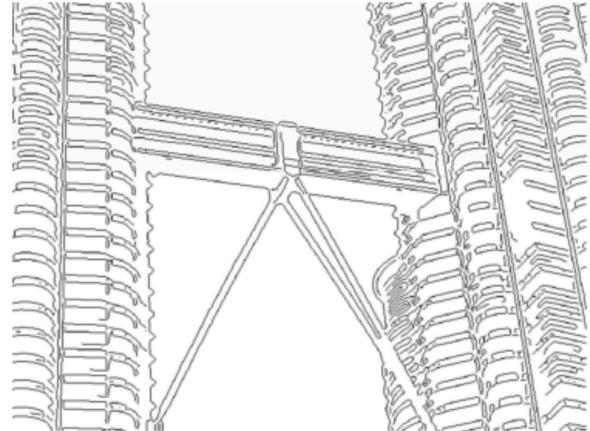


Figura 20: Immagine originale (a sinistra) e immagine con i bordi rilevati con il rilevatore di Canny (a destra)

Infine, la seconda ed ultima tipologia di rilevamento dei bordi è la trasformata di Hough, che sfrutta un metodo matematico per rilevare linee in un'immagine. Ciò, può essere usata per rilevare bordi potenzialmente sparsi, rotti od isolati; in linee utili, corrispondenti ai bordi dell'immagine. Tale processo consiste nel prendere un punto di coordinata (x, y) e memorizzare tutti i punti appartenenti alla retta $y = ax + b$. Ad esempio data il punto $(x, y) = (1, 3)$ si ottengono i punti alle coordinate $3 = a + b$, perciò $a = 3 - b$, come $(x, y) = (0, 3)$. Ripetendo ciò per gli altri punti, si ottengono una serie di punti che intersecano varie linee: i punti che hanno più intersezioni corrispondono alle linee più lunghe dell'immagine.

Tuttavia, le linee verticali hanno gradiente infinito: per risolvere ciò si usano le coordinate normali.

$$\rho = x \cos \theta + y \sin \theta$$

A questo punto, le linee più rilevanti dell'immagine, corrispondono ai valori più elevati di (ρ, θ) .

Capitolo 3:

Filtri nel dominio della frequenza

Fino ad ora, sono state trattate le immagini nel dominio spaziale, considerandole come funzioni a due variabili $f(x, y)$. Questa rappresentazione è sicuramente molto comoda ed intuitiva, poiché nelle immagini in bianco e nero, non è altro che una matrice $M \times N$ dove ogni elemento corrisponde al livello di grigio di un pixel. Tuttavia, tale rappresentazione diventa decisamente meno comoda nel momento in cui si eseguono delle operazioni di filtraggio, perché l'operazione di convoluzione risulta decisamente costosa in termini computazionali.

3.1: Dominio della frequenza

Per risolvere tale problema, si passa nel dominio della frequenza. L'idea di base, è:

- una funzione periodica si può riscrivere in serie di Fourier come somma di seni e coseni moltiplicati per coefficienti distinti;

$$f(t) = \sum_{n=-\infty}^{\infty} c_n \exp\left(j2\pi \frac{nt}{T}\right) : f(t) = f(t - T)$$

- una funzione non periodica si può rappresentare in frequenza come l'integrale di seni e coseni moltiplicati per una funzione pesata.

$$F(\nu) = \int_{-\infty}^{\infty} f(t) \exp(-j2\pi t\nu) dt$$

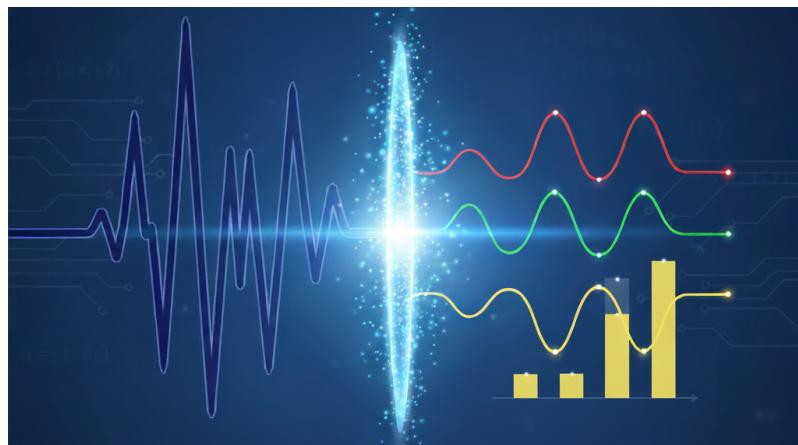


Figura 21: Rappresentazione della trasformata di Fourier

3.1.1: Trasformata di Fourier in due dimensioni

Passando in un due dimensioni, le frequenze sono due:

- frequenza spaziale lungo l'asse x , definita come u ;
- frequenza spaziale lungo l'asse y , definita come v .

Come si può intuire facilmente, se la funzione pesata in una dimensione ha due variabili, in due dimensione ne servono ben quattro: infatti x e y saranno gli indici di sommatoria, stessa cosa per l'antitrasformata che saranno u e v .

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \exp \left[-j2\pi \left(\frac{ux}{M} + \frac{vy}{N} \right) \right]$$

$$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u, v) \exp \left[j2\pi \left(\frac{ux}{M} + \frac{vy}{N} \right) \right]$$

Il valore della trasformata di Fourier alla frequenza di origine prende il nome di componente DC: esso si calcola facendo la media tra il prodotto dell'intensità di $f(x, y)$ e un fattore MN . Infatti, risulta molto comune eseguire una traslazione della componente DC sul punto $(M/2, N/2)$.

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f(x, y) + (1)^{x+y}] \exp \left[-j2\pi \left(\frac{ux}{M} + \frac{vy}{N} \right) \right]$$

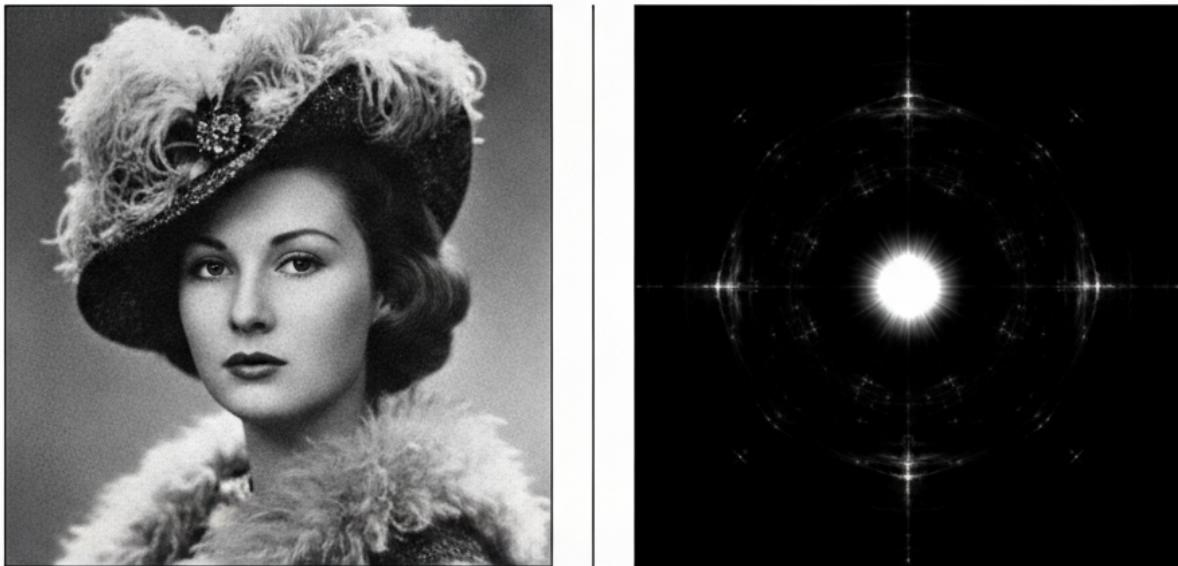


Figura 22: Immagine nel dominio spaziale (a sinistra) ed in frequenza (a destra)

3.1.2: Proprietà della trasformata di Fourier

La trasformata di Fourier gode delle seguenti proprietà:

- proprietà di traslazione, che afferma che se l'immagine è traslata nello spazio, in frequenza sarà traslata solo nella fase ma non in ampiezza;

$$F[f(x - x_0, y - y_0)] = F(u, v) \exp \left[-j2\pi \left(\frac{ux_0}{M} + \frac{vy_0}{N} \right) \right]$$

- linearità;

$$F[af_1(x, y) + bf_2(x, y)] = aF_1(u, v) + bF_2(u, v)$$

- proprietà di rotazione, che afferma che se l'immagine è ruotata di un angolo θ , anche in frequenza sarà ruotata dello stesso angolo;
- separabilità, ossia si può eseguire prima la trasformata sulle righe e poi sulle colonne (o viceversa).

3.1.3: Teorema della convoluzione e applicazione al filtraggio

La potenzialità del passaggio dal dominio spaziale al dominio della frequenza sta nel teorema della convoluzione. Tale teorema afferma che eseguire una convoluzione nel dominio spaziale corrisponde ad eseguire un prodotto nel dominio della frequenza, con un costo computazione molto meno elevato.

$$f(x, y) * h(x, y) \longleftrightarrow F(u, v)H(u, v)$$

Ciò consiste nel trasformare l'immagine originale in frequenza; moltiplicarla per il filtro in frequenza, denominato anche funzione di trasferimento; e trasformare l'immagine filtrata nel dominio spaziale. In particolare, si fa riferimento a due tipologie di frequenza:

- le basse frequenze, che rappresentano le aree omogenee;
- le alte frequenze, che rappresentano i dettagli.

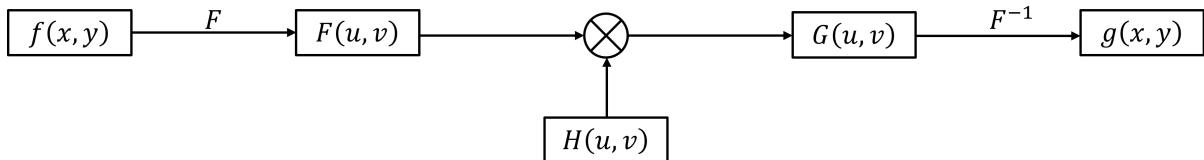


Figura 23: Filtraggio nel dominio della frequenza

3.2: Filtro di Notch

Il filtro più semplice da implementare è il filtro di Notch, chiamato anche filtro elimina-banda, che consiste nel far passare tutte le frequenze, eccetto per una tacca (notch in inglese), che corrisponde alla componente DC.

$$H(u, v) = \begin{cases} 0 & \text{se } (u, v) = \left(\frac{M}{2}, \frac{N}{2}\right) \\ 1 & \text{se } (u, v) \neq \left(\frac{M}{2}, \frac{N}{2}\right) \end{cases}$$

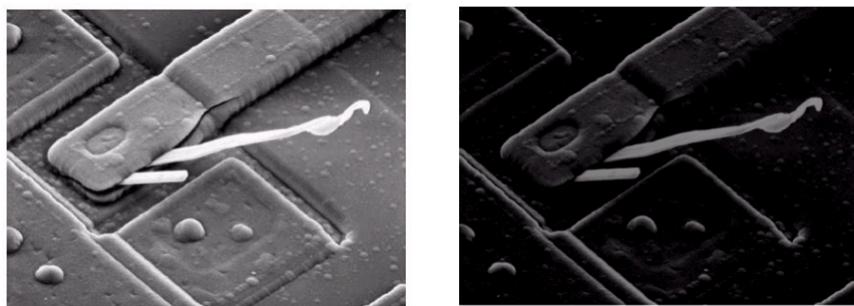


Figura 24: Immagine originale (a sinistra) e immagine filtrata con un filtro Notch (a destra)

3.3: Filtri passa-basso

I filtri passa-basso (LPF, dall'inglese low-pass filter) attenuano le alte frequenze, applicando quindi l'effetto di sfocatura e riduzione del rumore. Infatti, corrispondono ai filtri di smussamento nel dominio spaziale. Innanzitutto, si considera la distanza euclidea in frequenza $D(u, v)$, calcolata come segue.

$$D(u, v) = \sqrt{\left(u - \frac{M}{2}\right)^2 + \left(v - \frac{N}{2}\right)^2}$$

Inoltre, si considera la cosiddetta frequenza di cutoff (D_0), che cambia il comportamento del filtro in base a tale valore.

3.3.1: Filtro passa-basso ideale

Il filtro passa-basso ideale fa passare solamente le frequenze la cui distanza euclidea è minore od al più uguale alla frequenza di cutoff.

$$H(u, v) = \begin{cases} 1 & \text{se } D(u, v) \leq D_0 \\ 0 & \text{se } D(u, v) > D_0 \end{cases}$$

In particolare, maggiore sarà il cutoff e minore sarà la sfocatura, poiché passeranno più frequenze.

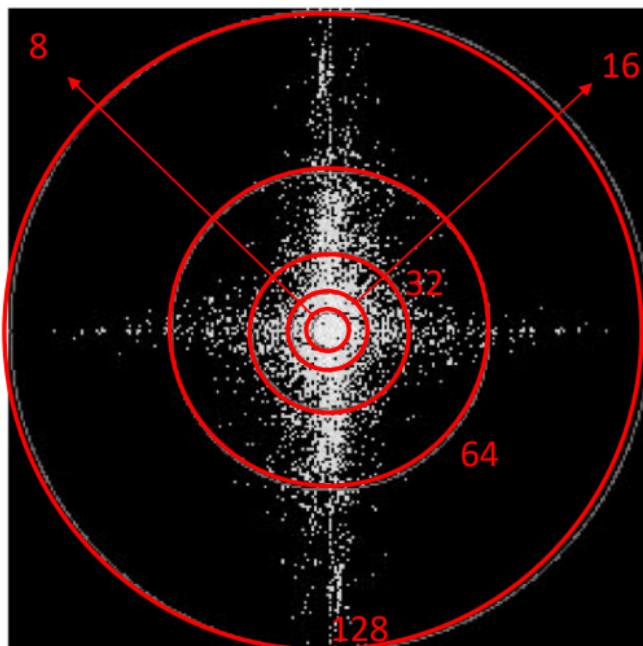


Figura 25: Funzionamento di un filtro passo-basso ideale

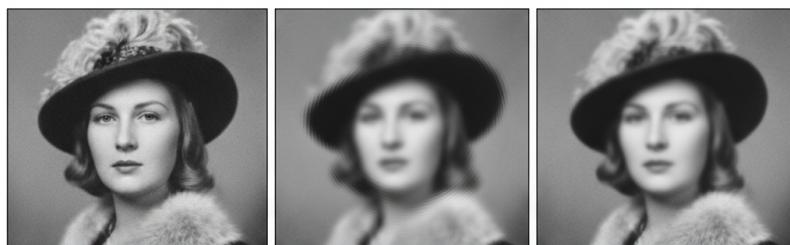


Figura 26: Da sinistra: immagine originale, LPF ideale $D_0 = 8$ e LPF ideale $D_0 = 16$

3.3.2: Filtro passa-basso di Butterworth

Il filtro passa-basso di Butterworth permette di eseguire una sfocatura molto meno intensa rispetto a quella ideale, ma neanche ai livello del filtro Gaussiano. Oltre alla distanza euclidea ed alla frequenza di cutoff, si aggiunge un parametro n , che indica l'ordine del filtro, ovvero la sua ripidità: infatti, maggiore è l'ordine, più si avvicina ad un filtro passa basso ideale.

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

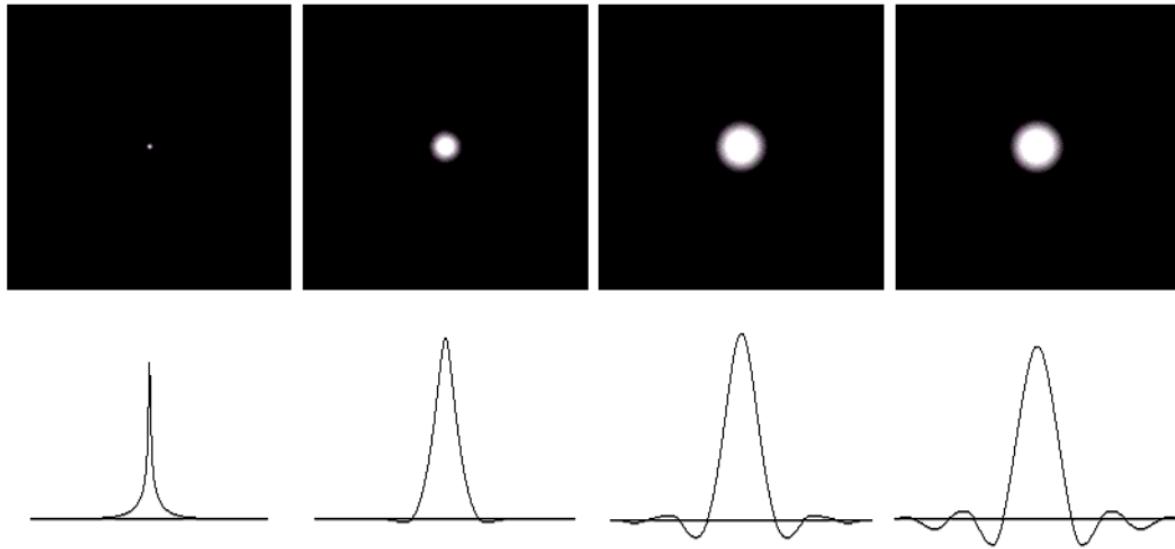


Figura 27: Vari ordini del filtro passa-basso di Butterworth

3.3.3: Filtro passa-basso Gaussiano

Il filtro passa-basso Guassiano è in grado di attenuare le alte frequenze con poca intensità. In particolare, il cutoff corrisponde alla deviazione standard della gaussiana.

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

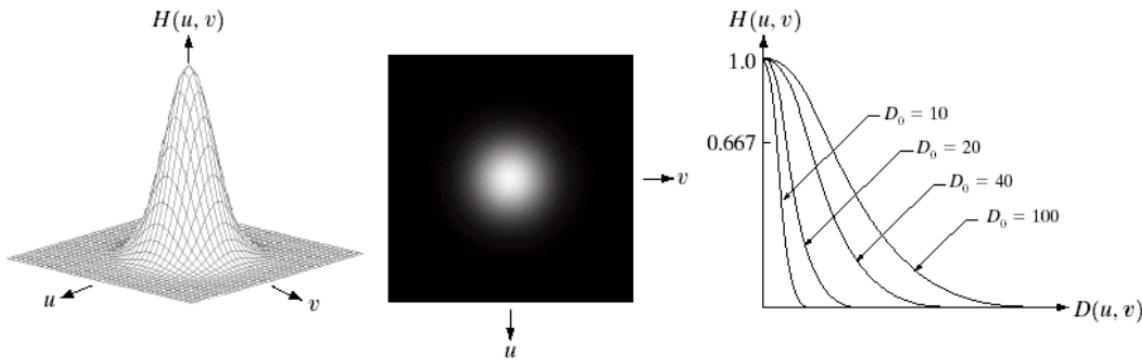


Figura 28: Filtri passa-basso Gaussiani

3.4: Filtri passa-alto

I filtri passa-alto (HPF, dall'inglese high-pass filter) attenuano le basse frequenze e risaltano le alte frequenze, applicando quindi l'effetto di sfocatura e riduzione del rumore. Infatti, corrispondono ai filtri di nitidezza nel dominio spaziale. Anche in questo caso, si ha la distanza euclidea e la frequenza di cutoff. Inoltre, la funzione di trasferimento di un filtro passo-alto è l'inverso di quello del filtro passa-basso.

$$H_{HPF}(u, v) = 1 - H_{LPF}(u, v)$$

3.4.1: Filtro passa-alto ideale, di Butterworth e Gaussiano

Di seguito, sono riportate le formule dei filtri passa-alto ideale, di Butterworth e Gaussiano, che presentano le proprietà inverse di quelle del passo-basso.

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



Figura 29: Da alto a sinistra senso orario: originale, HPF ideale, HPF di Butterworth e HPF Gaussiano

3.4.2: Il Laplaciano, i filtri di contrasto e l'high-boost filtering

Nel dominio spaziale, il Laplaciano presenta la formula seguente:

$$g(x, y) = f(x, y) - \nabla^2 f(x, y)$$

dove:

$$\nabla^2 f = \frac{\delta^2 f}{\delta x^2} + \frac{\delta^2 f}{\delta y^2}$$

In frequenza, la trasformata di una Fourier della derivata è la seguente:

$$F \left[\frac{d^2 f}{dt^2} \right] = -\nu^2 F(\nu)$$

che in due dimensioni e per le immagini, corrisponde:

$$F [\nabla^2 f] = -(u^2 + v^2) F(u, v)$$

Perciò, l'immagine trasformata in frequenza si ottiene nella maniera che segue:

$$G(u, v) = F(u, v) + (u^2 + v^2) F(u, v) = [1 + (u^2 + v^2)] F(u, v)$$

da cui si ricava la funzione di trasferimento.

$$H(u, v) = 1 + (u^2 + v^2)$$

Infine, per quanto riguarda i filtri di contrasto e l'high-boost filtering, le formule sono rispettivamente le seguenti.

$$H(u, v) = 1 - H_{lp}(u, v)$$

$$H(u, v) = (A - 1) + H_{lp}(u, v)$$

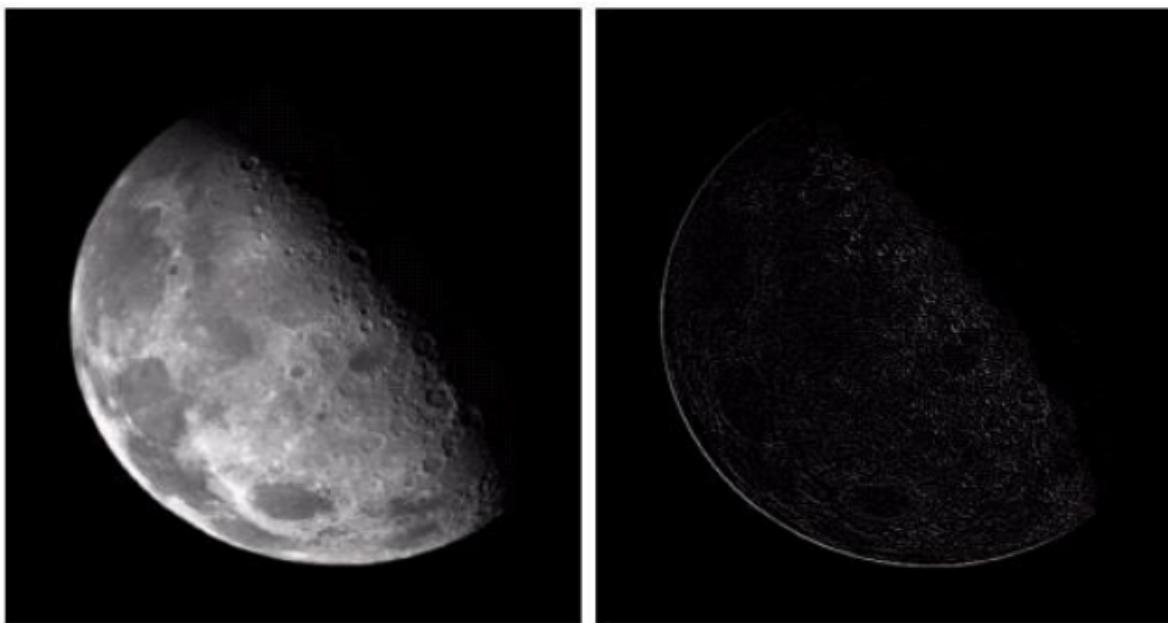


Figura 30: Immagine originale (a sinistra) e maschera del filtro Laplaciano (a destra)

Capitolo 4:

Trasformata Wavelet

In questo capitolo verranno mostrati i limiti della trasformata di Fourier, per poi introdurre la trasformata Wavelet, le sue potenzialità e le possibili applicazioni.

4.1: Limiti della trasformata di Fourier e possibile soluzione

Per comprendere bene i limiti della trasformata di Fourier, è necessario analizzare bene le caratteristiche di un'immagine.

4.1.1: Caratteristiche di un'immagine

Un'immagine si può vedere come un'insieme di regioni di texture e livelli di grigio simili, che combinati tra loro formano oggetti. In particolare, si considera:

- l'analisi a bassa risoluzione, che analizza gli oggetti ad alto contrasto, quindi le basse frequenze;
- l'analisi ad alta risoluzione, che analizza gli oggetti a basso contrasto, perciò le alte frequenze;
- l'analisi multirisoluzione, che analizza gli oggetti a contrasto variabile.

Da qui, si nota che l'immagine è un segnale non stazionario: per prima conviene analizzare la figura seguente. Come si può osservare:

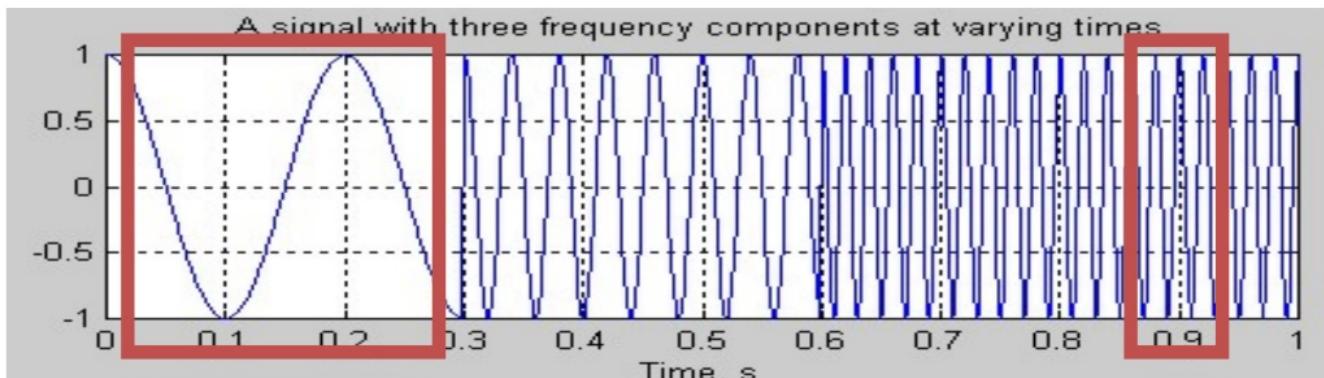


Figura 31: Variazione delle frequenze nell'asse temporale da $t_0 = 0\text{ s}$ a $t_3 = 1\text{ s}$

- da $t_0 = 0\text{ s}$ a $t_1 = 0,3\text{ s}$ si hanno le basse frequenze, in cui il segnale oscilla molto lentamente;
- da $t_1 = 0,3\text{ s}$ a $t_2 = 0,85\text{ s}$ si hanno le medie frequenze, dove le oscillazioni diventano sempre più ravvicinate e veloci;
- da $t_2 = 0,85\text{ s}$ a $t_3 = 1\text{ s}$ si hanno le alte frequenze, dove il segnale oscilla molto velocemente.

Perciò, le immagini hanno le statistiche che variano localmente: per questo motivo costruire un modello statistico che copre l'intera immagine, risulta decisamente molto complesso.

4.1.2: Problemi della trasformata di Fourier

Tornando alla figura precedente, se si eseguisse la trasformata di Fourier di quel segnale, essa darebbe come unica informazione la presenza di una bassa, media ed alta frequenza, senza fornire la loro posizione nel tempo. Generalizzando, eseguendo la trasformata di Fourier su un'immagine, si perdono completamente le informazioni temporali e spaziali.

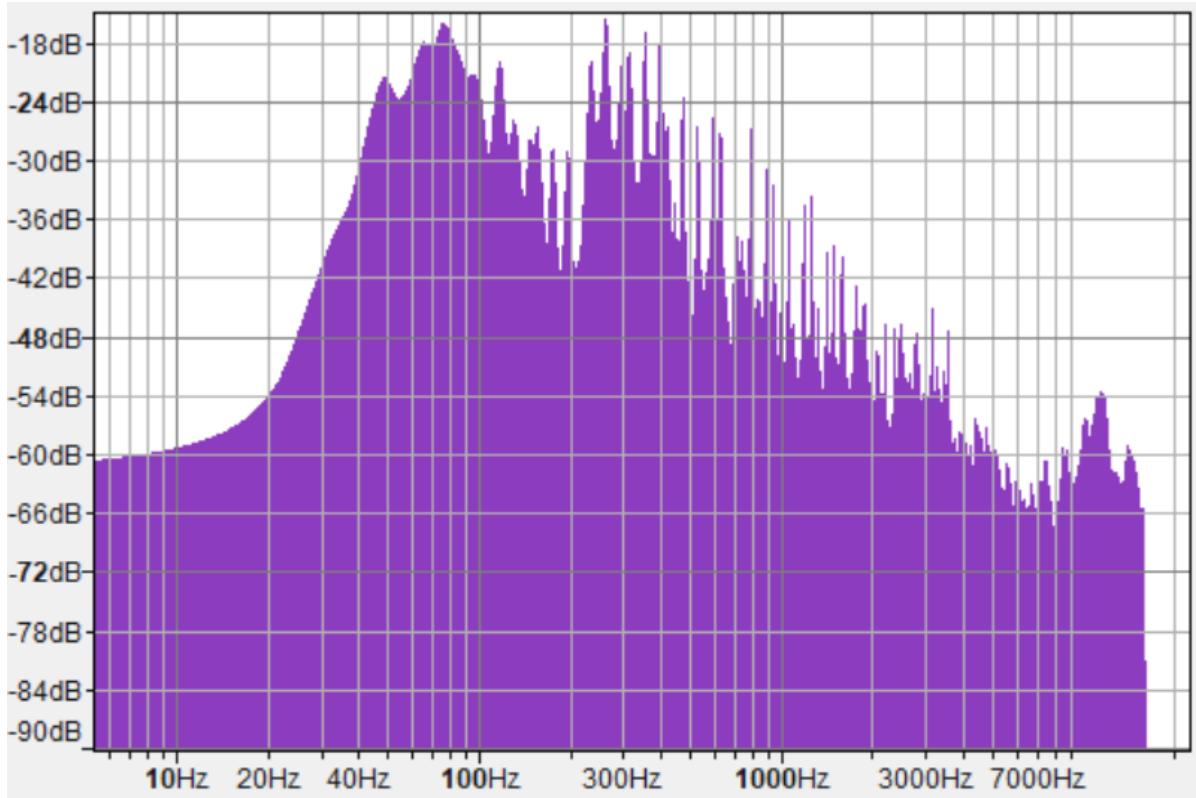


Figura 32: Rappresentazione dei problemi della trasformata di Fourier

4.1.3: Una soluzione temporanea: la trasformata di Fourier a breve termine

La trasformata di Fourier a breve termine (STFT, dall'inglese Short Time Fourier Transform) è una tecnica fondamentale per analizzare come il contenuto in frequenza di un segnale cambia nel tempo. Ciò consiste in:

1. si definisce una finestra di tempo di una dimensione fissa;
2. si applica la trasformata di Fourier al segnale finestrato;
3. si memorizzano le frequenze trovate in quel intervallo di tempo;
4. si scorre la finestra leggermente avanti nel tempo e si ripete le analisi, finché non finisce il segnale, costruendo una mappa.

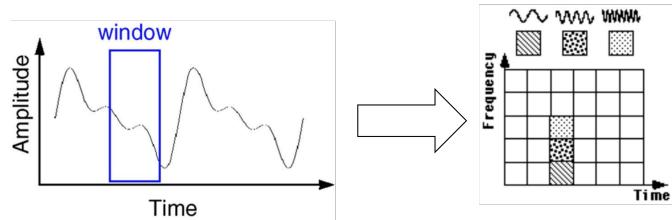


Figura 33: Esecuzione della STFT

Come si può facilmente intuire, la scelta della finestra risulta cruciale, poiché più la finestra è piccola, maggiore sarà la risoluzione temporale. Tuttavia, se si prende una finestra troppo piccola, si rischia che le frequenze rappresentabili siano insufficienti, tanto sarà debole il potenziale di discriminazione delle frequenze.

4.2: Introduzione alla trasformata Wavelet

La trasformata Wavelet si basa su piccole onde, dette appunto wavelet, che godono delle seguenti tre proprietà:

- frequenza ed ampiezza variabile;
- durata limitata, perciò non devono essere sinusoidi;
- valor medio nullo.

Di seguito, è riportata una tabella che mostra le wavelet più utilizzate.

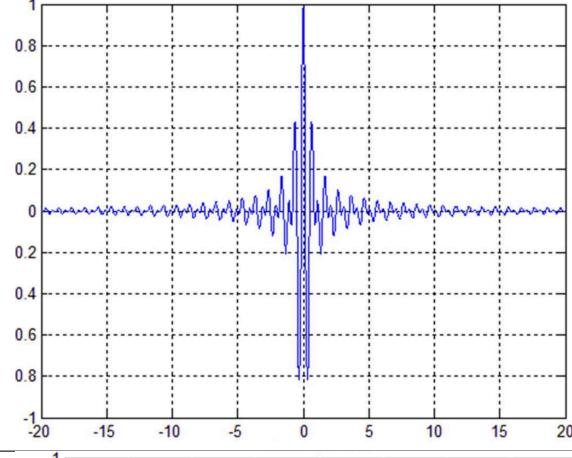
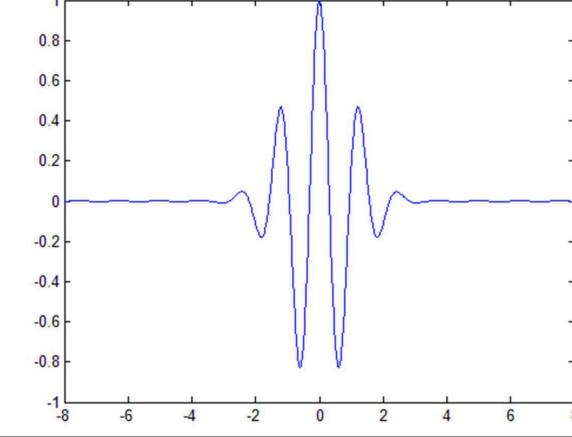
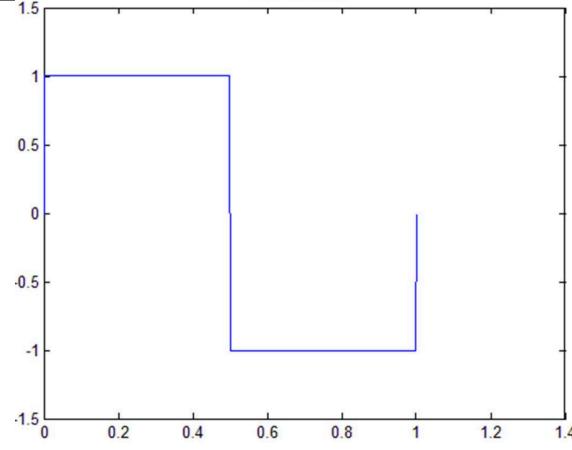
Nome	Formula analitica	Grafico
Shannon	$\psi(t) = 2 \operatorname{sinc}(2t) - \operatorname{sinc}(t)$	
Morlet	$\psi(t) = \exp\left(-\frac{t^2}{2}\right) \cos(5t)$	
Haar	$\psi(t) = \operatorname{rect}_{1/2}\left(t - \frac{1}{4}\right) - \operatorname{rect}_{1/2}\left(t - \frac{3}{4}\right)$	

Figura 34: Tipologie wavelet fondamentali

4.2.1: Determinazione della trasformata Wavelet

A questo punto, ogni segnale (non necessariamente periodico), può essere scritto in serie di wavelet.

$$f(t) = \sum_i a_i \psi_i(t)$$

$\psi_{s,\tau}$ è così definito:

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}} \psi\left(\frac{t-\tau}{s}\right)$$

dove:

- s è la scalatura;
- τ è la traslazione nel tempo;
- $\frac{1}{\sqrt{s}}$ è la normalizzazione;
- ψ è la wavelet madre;
- $\psi_{s,\tau}$ è la wavelet scalata e traslata nel tempo.

Adesso, si possono ricavare i coefficienti della trasformata Wavelet.

$$\gamma(s, \tau) = \int_{-\infty}^{\infty} f(t) \psi_{s,\tau}^*(t) dt$$

Una volta eseguito ciò, è possibile ricostruire il segnale con la formula seguente.

$$f(t) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \gamma(s, \tau) \psi_{s,\tau}(t) d\tau ds$$

Nella figura che segue, viene mostrata una rappresentazione di una possibile trasformata Wavelet, partendo da un segnale $f(t)$.

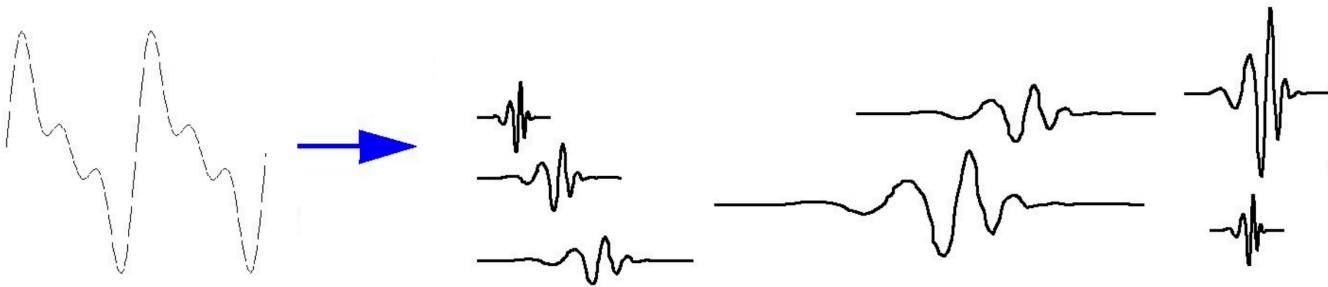


Figura 35: Rappresentazione di una trasformata Wavelet

4.2.2: Proprietà

La trasformata Wavelet gode delle seguenti proprietà:

- localizzazione simultanea spaziale e temporale, in cui la localizzazione della wavelet permette esplicitamente di rappresentare gli eventi nel tempo e la forma delle wavelet permettono di rappresentare i dettagli differenti e risoluzione differenti;
- sparsità, in cui le funzioni usate nella pratica presentano coefficienti pari a zero o molto piccoli;
- adattabilità, poiché si possono rappresentare funzioni discontinue e con angoli in modo molto efficiente;
- complessità temporale lineare, cioè diverse trasformazioni si possono compiere in tempo $O(N)$.

4.3: Analisi multirisoluzione

Per effettuare l'analisi multirisoluzione di un'immagine, è necessario definire in maniera compatta la formula dell'espansione di un segnale $f(t)$ in serie.

$$f(t) = \sum_k \sum_j a_{jk} \psi_{jk}(t) = \sum_k \sum_j a_{jk} 2^{\frac{j}{2}} \psi(2^j t - k)$$

i In particolare:

- a_{jk} sono coefficienti reali di espansione;
- $\psi_{jk}(t)$ sono le funzioni di espansione.

In particolare, si crea un mapping dove le k sono le ascisse e j le ordinate: le $\hat{f}_j(t)$ più in basso rappresentano i dettagli a bassa risoluzione; fino ad arrivare più vicino a $f(t)$, che rappresentano i dettagli ad alta risoluzione. Ne è mostrato un esempio nell'immagine di seguito.

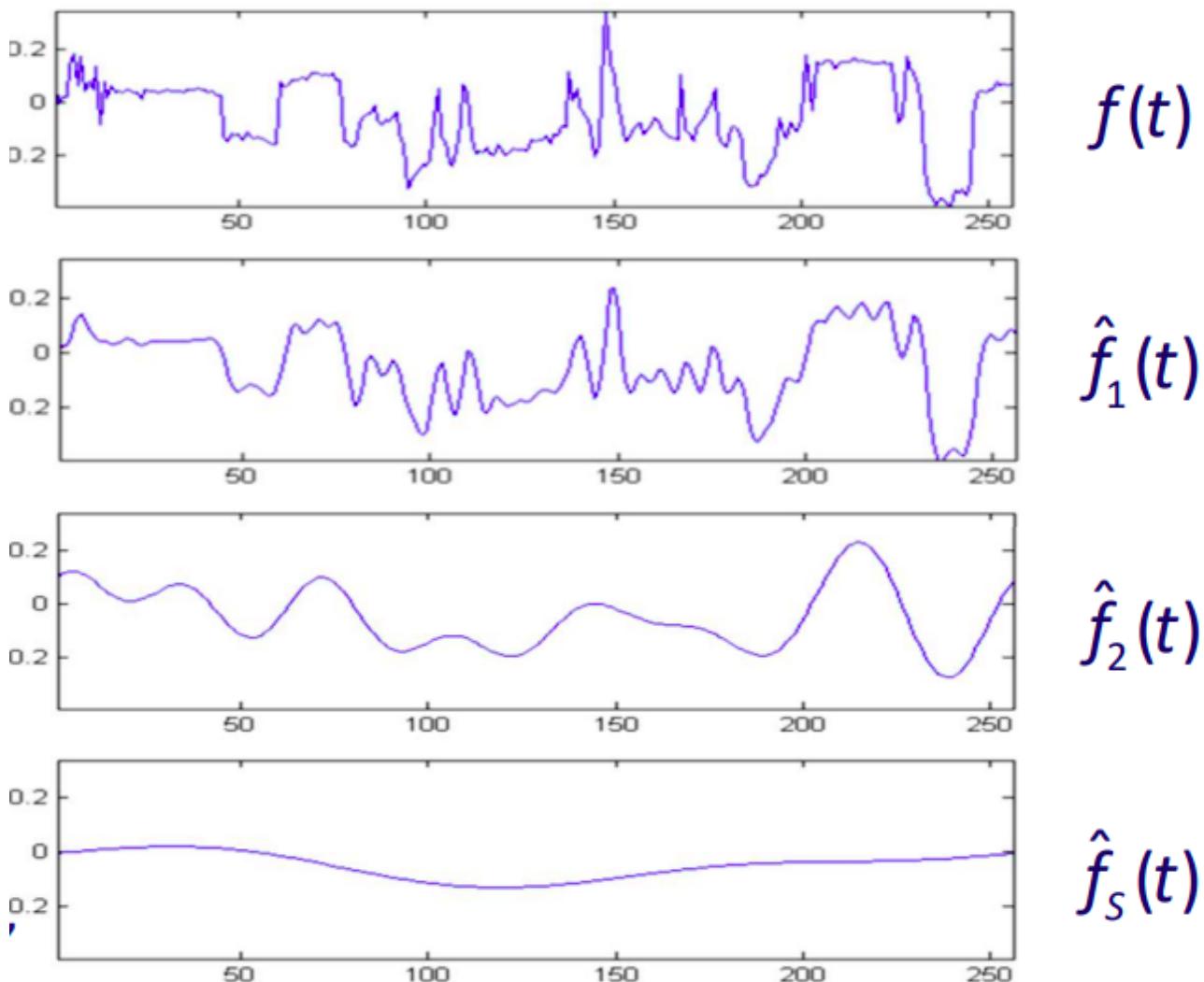


Figura 36: Analisi dei dettagli delle serie

4.3.1: Creazione delle piramide dell'immagine

Una semplice struttura per rappresentare un'immagine a più risoluzione è la cosiddetta piramide dell'immagine, in cui:

- alla base è presente la rappresentazione dell'immagine con la massima risoluzione, detto livello J;
- all'apice è presente la rappresentazione dell'immagine con la minima risoluzione, detto livello 0.

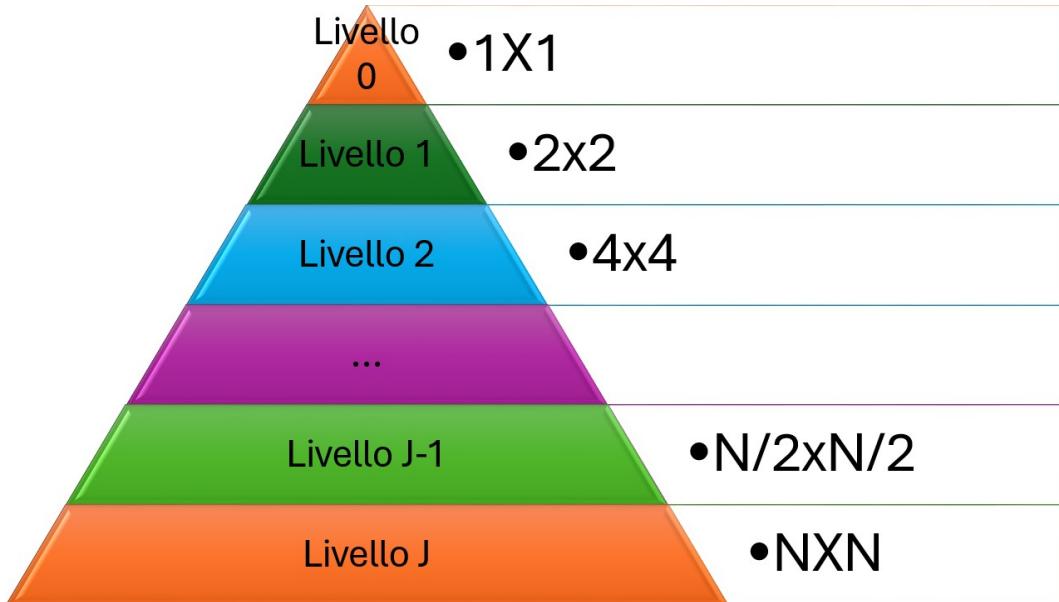


Figura 37: Piramide dell'immagine

La costruzione della piramide avviene nel modo seguente:

1. si riduce la risoluzione con un filtro di approssimazione (es. Gaussiano) e si esegue un sottocampionamento di un fattore 2;
2. si sovraccampiona di un fattore 2 il risultato ottenuto e si applica un filtro di interpolazione, creando una predizione con la stessa risoluzione dell'ingresso iniziale;
3. si esegue la differenza tra la predizione e l'input iniziale.

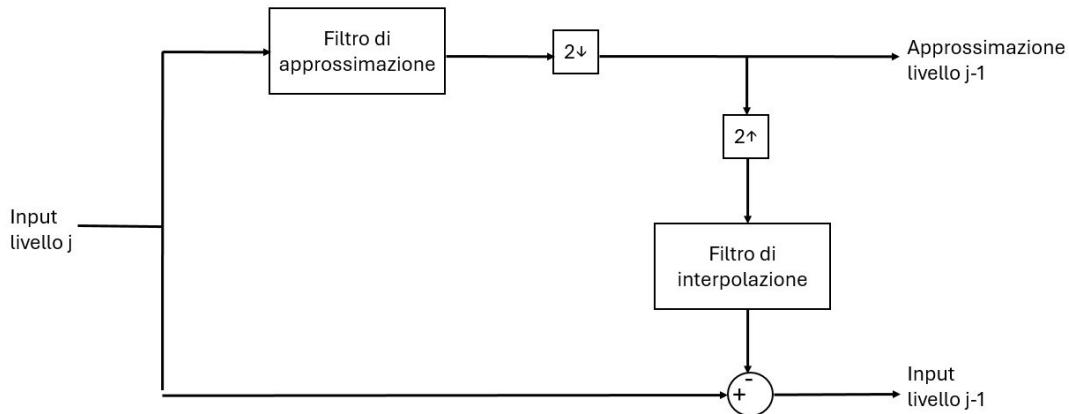


Figura 38: Schema di costruzione della piramide dell'immagine

4.3.2: Rappresentazione wavelet e condizioni

La rappresentazione wavelet consiste in un'approssimazione grossolana in totale dell'immagine, in cui viene influenzata dai coefficienti dei dettagli in scale differenti.

Se un insieme di basi di funzioni V si può rappresentare come una somma pesata di $\psi(2^j t - k)$, allora una gran parte dell'insieme (V incluso) si può rappresentare da una somma pesata di $\psi(2^{j+1} t - k)$. Quindi in questo caso, per $V_j \subseteq V_{j+1}$, se $f(t) \in V_j$ allora $f(t) \in V_{j+1}$.

A questo punto, l'idea di base è definire un insieme di basi di funzioni che coprono la differenza tra V_j e V_{j+1} : costruendo la base ortogonale W_j .

$$V_{j+1} = V_j + W_j$$

Infine, si sfruttano $\varphi(t)$ per V_j e $\psi(t)$ per W_j , per ricostruire la decomposizione come coppia di wavelet:

$$f(t) = \sum_k c_k \varphi(2^j t - k) + \sum_k d_{jk} \psi(2^j t - k)$$

dove:

- il primo fattore indicano le basse risoluzioni, dove $\varphi(t)$ è una funzione di scala per le basse risoluzioni;
- il secondo fattore indicano le alte risoluzioni.

4.4: Dalla trasformata di Haar alla trasformata Wavelet 2D

In questo paragrafo viene riportata la spiegazione da come si passa dalla trasformata Haar alla trasformata Wavelet in due dimensioni.

4.4.1: Trasformata Haar

La trasformata di Haar si basa su base di funzioni che sono molto semplici, soprattutto che hanno la proprietà di essere simmetriche, separabili ed esprimibili in forma matriciale.

$$\varphi(t) = \text{rect}\left(x - \frac{1}{2}\right) \quad \psi(t) = \text{rect}_{1/2}\left(x - \frac{1}{4}\right) - \text{rect}_{1/2}\left(x - \frac{3}{4}\right)$$

Considerando le formule seguenti di Wavelet padre (a sinistra) e Wavelet madre (a destra):

$$\varphi_{jk}(x) = 2^{\frac{j}{2}} \varphi(2^j x - k) \quad \psi_{jk}(x) = 2^{\frac{j}{2}} \psi(2^j x - k)$$

si considera un esempio di espansione in serie.

$$f(x) = \begin{cases} 8 & \text{se } 0 \leq x < 1/4 \\ 2 & \text{se } 1/4 \leq x < 1/2 \\ 4 & \text{se } 1/2 \leq x < 3/4 \\ 6 & \text{se } 3/4 \leq x < 1 \end{cases}$$

Per prima cosa, si espande in serie in approssimazione in alta risoluzione per $j = 2$.

$$\varphi_{2,k}(x) = 2\varphi(4x - k) = 2 \text{rect}(4x - k - \frac{1}{2}) = 2 \text{ (altezza 2)}$$

- $k = 0$: $8c_{2,0} = 2 \leftrightarrow c_{2,0} = 4$;
- $k = 1$: $2c_{2,1} = 2 \leftrightarrow c_{2,1} = 1$;
- $k = 2$: $4c_{2,2} = 2 \leftrightarrow c_{2,2} = 2$;
- $k = 3$: $6c_{2,3} = 2 \leftrightarrow c_{2,1} = 3$.

$$f(x) = 4\varphi_{2,0}(x) + \varphi_{2,1}(x) + 2\varphi_{2,2}(x) + 6\varphi_{2,3}(x)$$

Tuttavia, tale struttura non dice nulla sulla struttura dell'approssimazione e sul dettaglio, perciò si riscrive la formula $V_2 = V_1 + W_0 + W_1 = V_0 + W_0 + W_1$.

- $V_0 \rightarrow \varphi_{0,0}$;
- $W_0 \rightarrow \psi_{0,0}$;
- $W_1 \rightarrow \psi_{1,0}, \psi_{1,1}$.

Per quanto riguarda i coefficienti, si calcolano nel modo seguente.

$$c_{j,k} = \frac{A+B}{\sqrt{2}} \quad d_{j,k} = \frac{A-B}{\sqrt{2}}$$

Da $V_2 = V_1 + W_1$:

- $c_{1,0} = \frac{c_{2,0}+c_{2,0}}{\sqrt{2}} = \frac{4+1}{\sqrt{2}} = \frac{5}{\sqrt{2}}$ (approssimazione);
- $c_{1,1} = \frac{c_{2,2}+c_{2,3}}{\sqrt{2}} = \frac{2+3}{\sqrt{2}} = \frac{5}{\sqrt{2}}$ (approssimazione);
- $d_{1,0} = \frac{c_{2,0}-c_{2,0}}{\sqrt{2}} = \frac{4-1}{\sqrt{2}} = \frac{3}{\sqrt{2}}$ (dettaglio);
- $d_{1,1} = \frac{c_{2,2}-c_{2,3}}{\sqrt{2}} = \frac{2-3}{\sqrt{2}} = -\frac{1}{\sqrt{2}}$ (dettaglio).

Da $V_1 = V_0 + W_0$:

- $c_{0,0} = \frac{c_{1,0}+c_{1,1}}{\sqrt{2}} = (\frac{5}{\sqrt{2}} + \frac{5}{\sqrt{2}}) \frac{1}{\sqrt{2}} = 5$;
- $d_{0,0} = \frac{c_{1,0}-c_{1,1}}{\sqrt{2}} = (\frac{5}{\sqrt{2}} - \frac{5}{\sqrt{2}}) \frac{1}{\sqrt{2}} = 0$.

Quindi la serie della multirisoluzione è la seguente:

$$f(x) = 5\varphi_{0,0}(x) + 0\psi_{0,0}(x) + \frac{3}{\sqrt{2}}\psi_{1,0}(x) - \frac{1}{\sqrt{2}}\psi_{1,1}(x) = 5\varphi_{0,0}(x) + \frac{3}{\sqrt{2}}\psi_{1,0}(x) - \frac{1}{\sqrt{2}}\psi_{1,1}(x)$$

4.4.2: Codifica di sottobanda

La codifica di sottobanda consiste nel calcolare i coefficienti della bassa risoluzione da quelli di alta risoluzione, attraverso un algoritmo a struttura ad albero: in particolare si sfrutta un filtro passa-basso ed un filtro passa-alto. Ciò si applica sia alle righe che alle colonne, come mostrato di seguito. In particolare:

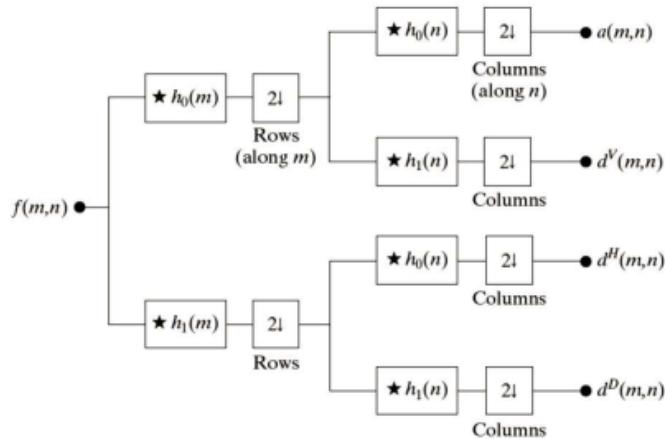


Figura 39: Processo di codifica di sottobanda

- $a(m, n)$ rappresenta l'approssimazione;
- $d^V(m, n)$ rappresentano i dettagli verticali di sottobanda;
- $d^H(m, n)$ rappresentano i dettagli orizzontali di sottobanda;
- $d^D(m, n)$ rappresentano i dettagli diagonali di sottobanda.

Ovviamente, una o più di queste sottobande può a loro volta essere suddivise in quattro sottobande più piccole e così via.

4.4.3: Vantaggi della trasformata di Haar e trasformata Wavelet 2D

La trasformata di Haar presenta i seguenti vantaggi:

- contiene i stessi pixel dell'immagine originale;
- le sue statistiche locali rimangono relativamente costanti e facilmente modellate;
- molti valori sono nulli od infinitesimi, perciò è un'ottima candidata alla compressione (trattata nel prossimo capitolo);
- si possono estrarre sia le approssimazioni grossolane sia quelle fini.

Per questo motivo, si estende ciò alla trasformata Wavelet in due dimensioni, dove i filtri passa-basso e passa-alto sono definiti rispettivamente con φ e ψ :

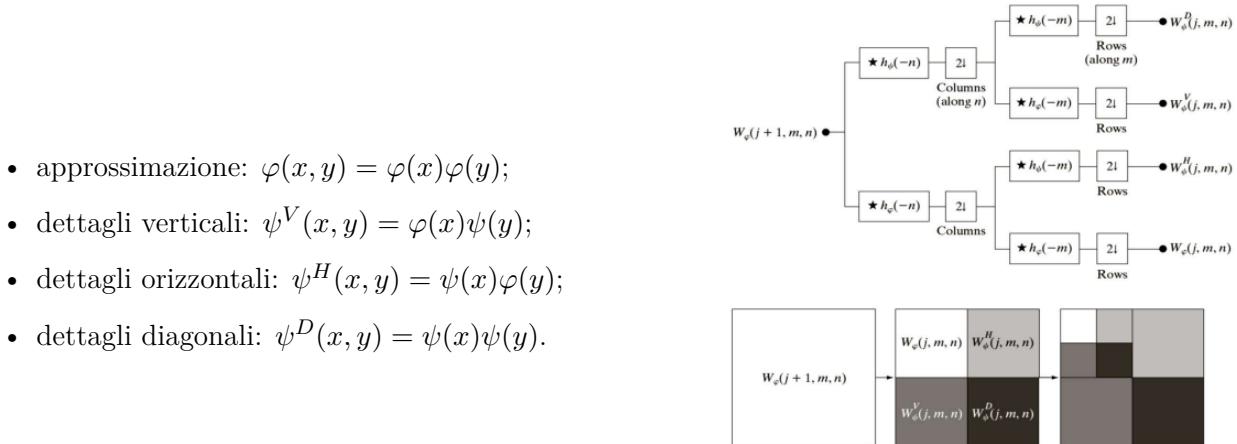


Figura 40: Esempio di trasformata Wavelet 2D

4.4.4: Elaborazione dell'immagini nel dominio wavelet

Nel dominio wavelet, l'elaborazione delle immagine avviene nel modo seguente:

1. decomporre l'immagine nel dominio walet scegliendo una wavelet madre (solitamente Haar) e il numero di livello di scala di decomposizione;
2. modificare i coefficienti wavelet a seconda della tipologia di elaborazione, come riduzione del rumore e compressione;
3. ricostruire l'immagine con i coefficienti wavelet modificati.

Capitolo 5:

Processo di compressione delle immagini

Il processo di compressione di un’immagine, consiste nel ridurre drasticamente la quantità di dati presenti nell’immagine, poiché trasmettere un’immagine non compressa richiede molto tempo e molta larghezza di banda, come mostrato nell’immagine seguente. Basti pensare che già un’immagine 1024×1024 in scala di

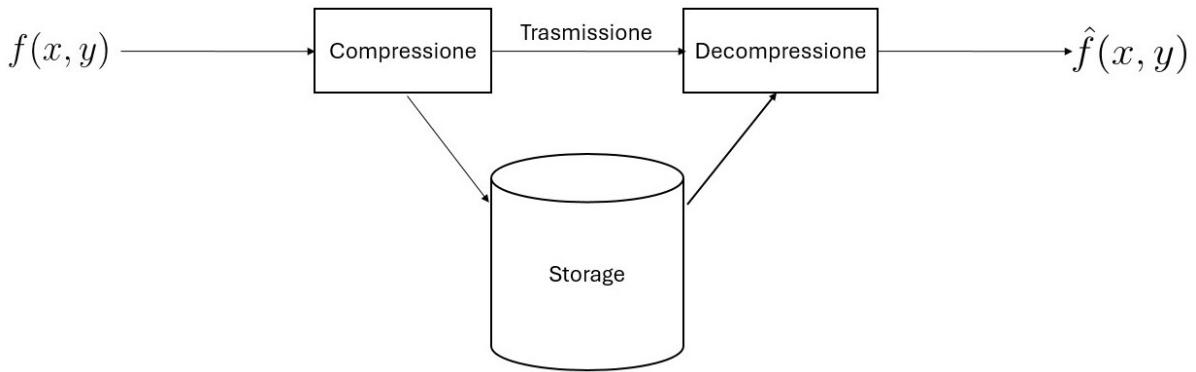


Figura 41: Processo di trasmissione di un’immagine compressa

grigi con risoluzione bassa ($8 b/p$), per essere trasmessa in 2G ($v = 50 kb/s$), sono necessari:

$$c = 1024 \times 1024 \times 8 = 8388608 b \quad b = \frac{c}{v} = \frac{8388608}{50000} = 167,77216 s \approx 3'$$

che è decisamente troppo elevato.

5.1: Ridondanza dei dati

Per prima cosa è necessario ribadire che dato ed informazione non sono assolutamente sinonimi. Infatti:

- un’informazione è ciò che si vuole trasmettere;
- un dato rappresenta i modi di trasmettere l’informazione.

La ridondanza dei dati si può quantificare matematicamente come entità. Infatti, si considerano le seguenti grandezze:

- rapporto di compressione (C_R), che è il rapporto tra due insiemi di dati che contengono la stessa informazione;

$$C_R = \frac{n_1}{n_2}$$

- ridondanza relativa dei dati (R_D), che indica la quantità dei dati che sono ridondanti, spesso indicati in percentuale.

$$R_D = 1 - \frac{1}{C_R}$$

In questo paragrafo, verranno trattate le diverse tipologie di rindondanza.

5.1.1: Ridondanza di codifica

La ridondanza di codifica si ha nel momento in cui si sceglie un sistema di codifica non efficiente. Per calcolare l'efficienza di un algoritmo di compressione, dipende dal livello medio di grigio, che dipende dalla probabilità che ogni livello di grigio dell'immagine sia presente in tale immagine e quanti bit occupa ($l(r_k)$).

$$p_r(r_k) = \frac{n_k}{MN} \quad L_{avg} = \sum_0^{L-1} l(r_k)p_r(r_k)$$

Per esempio data un'immagine 256×256 con le seguenti caratteristiche:

r_k	$p_k(r_k)$
$r_{87} = 87$	$3/10$
$r_{128} = 128$	$1/2$
$r_{186} = 186$	$1/10$
$r_{255} = 255$	$1/10$
$r_k \text{ se } k \neq 87, 128, 186, 255$	0

per esempio si usa per ogni livello di grigio il numero massimo rappresentabile del livello di grigio più alto (255), che è 8 ($\log_2(255)$): Il livello medio di grigio è $L_{avg_1} = 8b$. A questo punto conviene assegnare il numero di bit a seconda della probabilità: maggiore è la probabilità, meno bit deve avere quel livello di grigio, così lo spazio occupato sarebbe decisamente minore.

r_k	$p_k(r_k)$	$l_k(r_k)$
$r_{87} = 87$	$3/10$	2
$r_{128} = 128$	$1/2$	1
$r_{186} = 186$	$1/10$	3
$r_{255} = 255$	$1/10$	3

$$L_{avg_2} = \frac{3}{10} \times 2 + \frac{1}{2} \times 1 + \frac{1}{10} \times 3 + \frac{1}{10} \times 3 = \frac{17}{10} b = 1,7b$$

Infine, si calcolano le due grandezze.

$$C_R = \frac{8 \times 10}{17} = \frac{80}{17}$$

$$R_D = 1 - \frac{17}{80} = \frac{63}{80} = 78,75\%$$

5.1.2: Ridondanza interpixel

La ridondanza interpixel implica che qualsiasi valore di un pixel può essere predetto dai suoi vicini, grazie alla correlazione. Per ridurre ciò, i dati dovrebbero essere mappati.

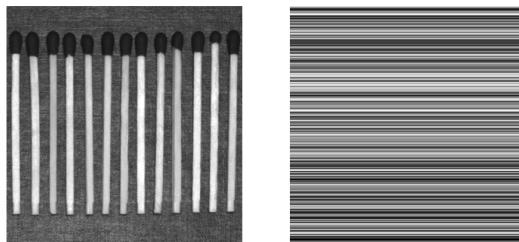


Figura 42: Esempio di ridondanza di interpixel

5.1.3: Ridondanza psicovisiva

La ridondanza psicovisiva sta nel fatto che alcuni pixel sono talmente simili che l'occhio umano non li percepisce, dato che il sistema visivo umano non percepisce tutta l'informazione visiva con la stessa intensità: ma cerca solo caratteristiche importanti, come angoli e texture. Il classico esempio è un'immagine a tinta unita.

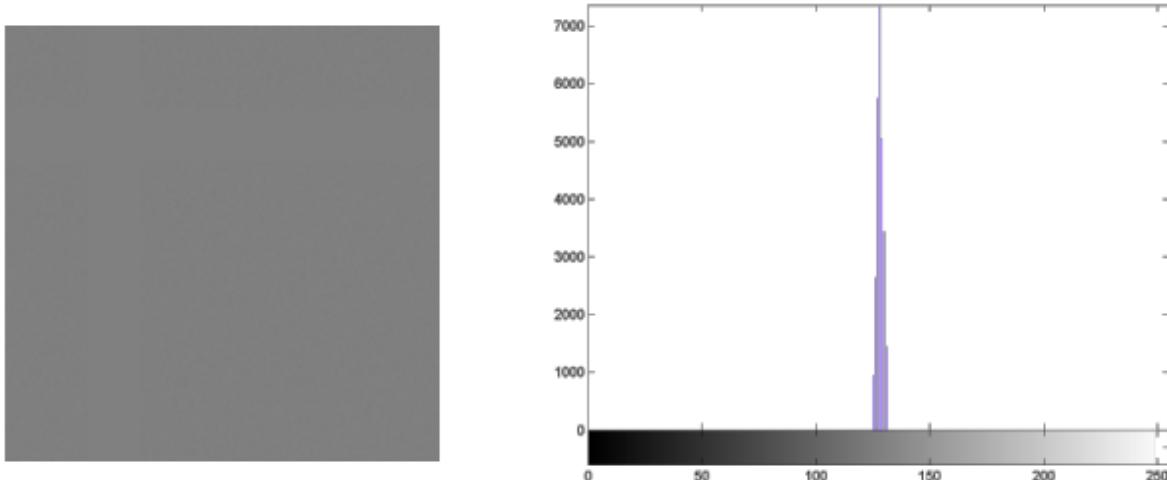


Figura 43: Esempio di ridondanza psicovisiva

5.2: Teoria dell'informazione

Per effettuare una codifica efficiente, è necessario conoscere dei cenni di teoria dell'informazione.

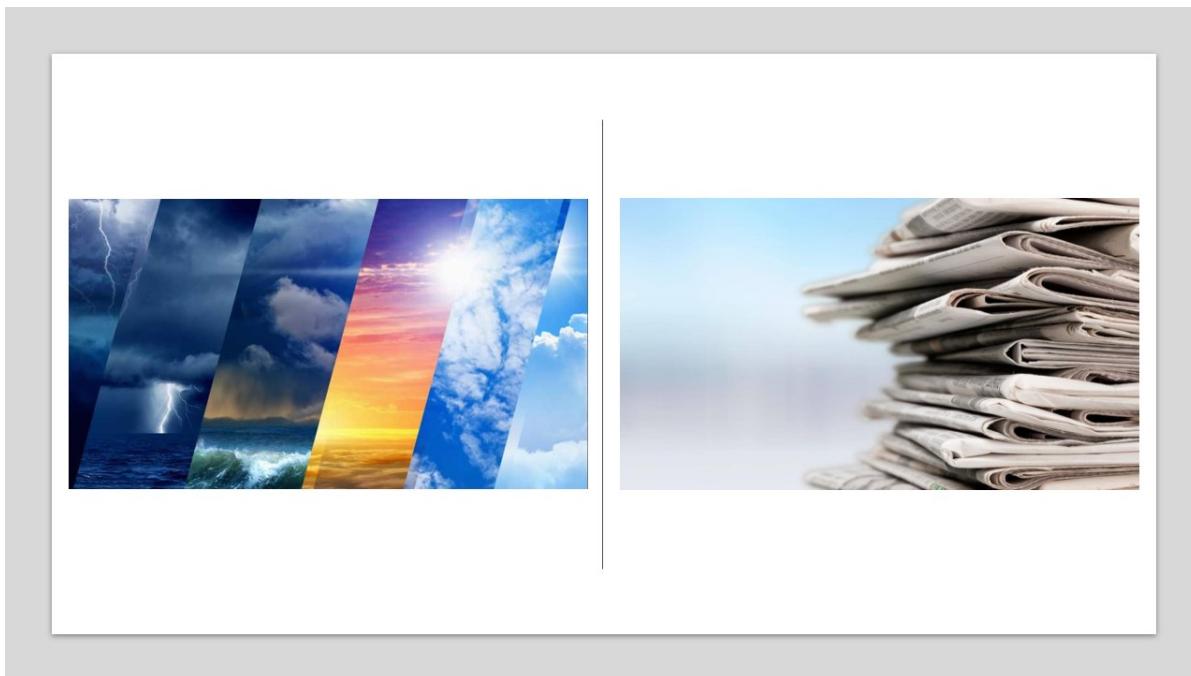


Figura 44: Esempio di rappresentazione della teoria dell'informazione

5.2.1: Concetti base di informazione

Per prima cosa è necessario comprendere i concetti base dell'informazione:

- più la probabilità di un evento è bassa più il contenuto informativo è alto;
- l'informazione di più messaggi indipendenti sono la somma delle informazione corrispondenti.

Detto ciò, sia X una sorgente che genera un messaggio x_i con probabilità $p(x_i)$, l'informazione si calcola nel modo seguente:

$$I(x_i) = \log_c \left[\frac{1}{p(x_i)} \right]$$

dove c è la base del logaritmo, che dipende dal tipo di codifica. Le più comuni sono il bit ($c = 2$) ed il nat ($c = e$).

5.2.2: Entropia

Un altro concetto fondamentale è l'entropia dell'informazione, che misura l'incertezza media e l'informazione attesa della sorgente.

$$H(X) = - \sum_{i=1}^N p(x_i) \log_c [p(x_i)]$$

L'entropia dell'informazione è protagonista del teorema della codifica di sorgente di Shannon, che afferma che è impossibile comprimere i dati che presentano come velocità di codifica, cioè il numero di simboli trasmessi per secondo, minore dell'entropia senza perdere del contenuto informativo.

5.2.3: Criteri di fedeltà

Infine, l'ultima parte analizzata della teoria dell'informazione è quella dedicata ai criteri di fedeltà, che quantifica quanta informazione è possibile perdere mantenendo ancora l'informazione accettata oppure no. In particolare, si considerano due tipologie di criteri di fedeltà.

Il primo criterio è quello soggettivo, che si basano solamente sulla comparazione visiva tra due immagini, classificando le immagini in ranking da eccellente ad inutilizzabile.

Il secondo ed ultimo criterio è quello oggettivo, che si basa in base a delle metriche matematiche, che dipendono dall'errore ($\epsilon(x, y)$) e dall'errore assoluto (ϵ), calcolati in base all'immagine originale ($f(x, y)$) ed all'immagine compressa ($\hat{f}(x, y)$).

$$\epsilon(x, y) = \hat{f}(x, y) - f(x, y) \quad \epsilon = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]$$

A questo punto, i criteri di fedeltà sono dettati dalle metriche di errore quadratico medio ($RMSE$), il rapporto segnale-rumore quadrico medio (SNR_{ms}) ed il rapporto segnale-rumore di picco ($PSNR$), dove questi ultimi due si misurano in Decibel (dB), che è una scala logaritmica.

$$RMSE = \sqrt{\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2}$$

$$SNR_{ms} = \sum_{y=0}^{N-1} \hat{f}^2(x, y) \left\{ \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2 \right\}^{-1}$$

$$PSNR = L_{max} \left\{ \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2 \right\}^{-1}$$

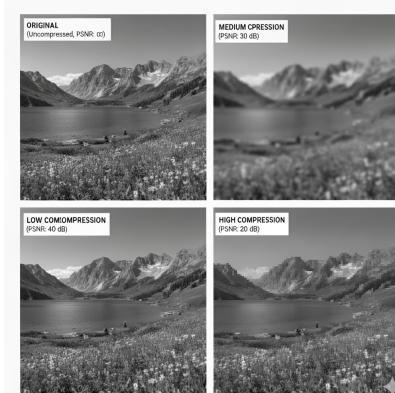


Figura 45: Misura del PSNR in un'immagine

5.3: Modello di compressione delle immagini

Il modello di compressione delle immagini consistono nei seguenti operazioni:

1. mappatura, che consiste nel prendere l'immagine ed eseguire una mappatura dei dati, riducendo la ridondanza interpixel (operazione reversibile);
2. quantizzazione, che esegue l'operazione di quantizzazione, riducendo la ridondanza psicovisiva (operazione irreversibile);
3. codifica di simbolo, che esegue un sistema di codifica dell'immagine (operazione reversibile) e lo invia al canale;
4. decodifica di simbolo, che dal canale prende l'immagine codifica e la ritrasforma prima della codifica;
5. mappatura inversa, che la riporta come matrice di pixel, leggermente diversa a causa del quantizzatore.

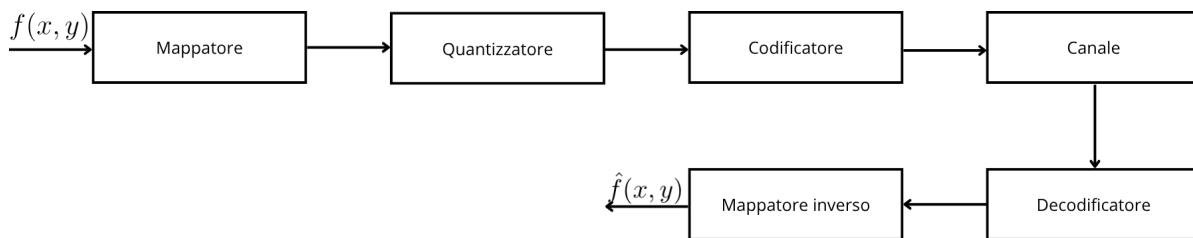


Figura 46: Schema del modello di compressione di un'immagine

A seconda dell'errore, la tipologia di compressione di un'immagine si suddividono in due grandi categorie:

- compressione lossless, in cui l'errore è nullo, perciò non viene perso alcuna informazione, sfruttando la ridondanza di codifica ed interpixel;
- compressione lossy, invece, sfrutta ogni tipologia di ridondanza (anche quella psicovisiva), tollerando qualche errore o qualche perdita d'informazione.

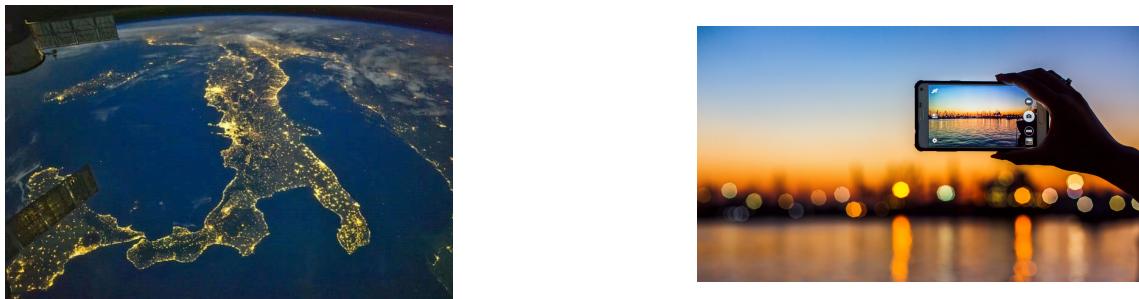


Figura 47: Esempio tipologia di immagine da comprimere lossless (a sinistra) od anche lossy (a destra)

In questo paragrafo, sono elencate la maggior parte dei sistemi di codifica più utilizzati.

5.3.1: Codifica di Huffmann

La codifica di Huffmann è un sistema di codifica lossless, che usa gli alberi binari come struttura dati per costruire la codifica, la cui costruzione si basa sulla probabilità di simbolo. Infatti, essa consiste in:

1. ordinare i simboli per probabilità;
2. combinare le due probabilità minori;
3. ripetere ciò, finché rimangono due probabilità.

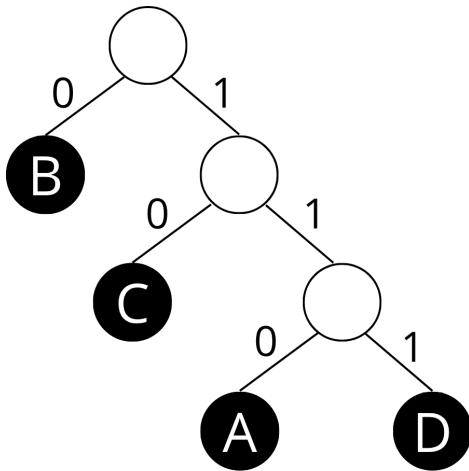


Figura 48: Esempio di una codifica di Huffman

Ad esempio, dato l'insieme $X = A, B, C, D$ con probabilità $p(X) = \{\frac{1}{8}, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}\}$, si ordina $X = B, C, A, D$ e si costruisce a partire da B e poi CAD , che a sua volta si divide in C e AD , che infine si divide in A e D . In particolare, Ogni volta che si va a sinistra si assegna uno 0, mentre 1 quando si va a destra. In particolare, nessun simbolo possiede il prefisso di un altro simbolo. Infatti:

- $A \rightarrow 110$;
- $B \rightarrow 0$;
- $C \rightarrow 10$;
- $D \rightarrow 111$.

5.3.2: Codifica aritmetica

La codifica aritmetica è una codifica lossless in cui non c'è una corrispondenza uno ad uno tra sorgente e codice e non è presente nessuna ipotesi che la codifica di simbolo avvenga uno alla volta. A questo punto viene generato un intervallo $[0; 1]$, in cui inizialmente viene suddiviso in base alla probabilità di simbolo. Poi, più il messaggio diventa lungo, minore sarà la distanza tra i due estremi dell'intervallo, in base alla probabilità.

Ad esempio, dati i seguenti simboli e le probabilità seguenti: $X = \{A, B, C\}$ $p(X) = \{0, 4; 0, 5; 0, 1\}$. Nel momento in cui si vuole ricostruire il messaggio BBC , si considerano i seguenti simboli:

1. simbolo B (intervallo di riferimento $[0; 1]$):
 - $A \rightarrow [0; 0, 4[$
 - $A \rightarrow [0, 4; 0, 9[$ (intervallo da considerare)
 - $C \rightarrow [0, 9; 1[$
2. simbolo B (intervallo di riferimento $[0, 4; 0, 9[$, $\Delta = 0, 9 - 0, 4 = 0, 5$):
 - $A \rightarrow 0, 4 \times 0, 5 = 0, 2 \rightarrow [0, 4; 0, 4 + 0, 2[\rightarrow [0, 4; 0, 6[$
 - $A \rightarrow 0, 5 \times 0, 5 = 0, 25 \rightarrow [0, 6; 0, 6 + 0, 25[\rightarrow [0, 6; 0, 85[$ (intervallo da considerare)
 - $C \rightarrow 0, 1 \times 0, 5 = 0, 05 \rightarrow [0, 85; 0, 85 + 0, 05[\rightarrow [0, 85; 0, 9[$
3. simbolo C (intervallo di riferimento $[0, 6; 0, 85[$, $\Delta = 0, 85 - 0, 6 = 0, 25$):
 - $A \rightarrow 0, 4 \times 0, 25 = 0, 1 \rightarrow [0, 6; 0, 6 + 0, 1[\rightarrow [0, 6; 0, 7[$
 - $A \rightarrow 0, 5 \times 0, 25 = 0, 125 \rightarrow [0, 7; 0, 7 + 0, 125[\rightarrow [0, 6; 0, 825[$
 - $C \rightarrow 0, 1 \times 0, 25 = 0, 025 \rightarrow [0, 825; 0, 825 + 0, 025[\rightarrow [0, 825; 0, 85[$ (intervallo da considerare)
4. fine: $[0, 825; 0, 85[$.

5.3.3: Codifica RLC

La codifica RLC (Run-length coding) è una codifica lossless molto semplice in cui concatena ogni simbolo con il numero di volte che si ripete consecutivamente. Ad esempio $AAABBAC$ diventa $(A, 3)(B, 2)(A, 1), (C, 1)$. Tuttavia, tale sistema di codifica non è molto efficiente, tuttavia è veramente semplice da implementare.

5.3.4: Codifica di Lempel Ziv

La codifica di Lempel Ziv è una codifica lossless che è un sistema di codifica che è composto da tre elementi: $< P, L, C >$. Dove:

- P indica quanti passi indietro bisognerebbe bloccare il testo decodificato;
- L è la lunghezza della stringa;
- C è il prossimo carattere della stringa.

A questo punto, l'algoritmo consiste in:

1. si decompone la sequenza d'ingresso in stringhe;
2. ogni volta che un blocco differisce dal precedente, viene inserito nel dizionario;
3. tutte le stringhe incluse nel dizionario, vengono associate ad una posizione;
4. nella procedura di codifica, l'algoritmo registra ogni stringa nuova nel dizionario e la sua posizione.

Per esempio, data la stringa $xyxxxyxyxxyy$, si procede come segue:

1. legge x : non disponibile nel dizionario, quindi aggiunge $< 0, 0, x >$;
2. (buffer $[x]$) legge y : non disponibile nel dizionario, quindi aggiunge $< 0, 0, y >$;
3. (buffer $[xy]$) legge x , che si trova a 2 posizioni indietro, si trova un match lungo 1 (il carattere prima della x è un'altra x), $< 2, 1, x >$;
4. (buffer $[xyxx]$) legge y , che si trova a 3 posizioni indietro, si trova un match lungo 2 (yx c'è, yxy non c'è), $< 3, 2, y >$;
5. (buffer $[xyxxxy]$) legge x , si cerca il match più lungo, che è xxy (3), che si trova 5 posizioni indietro, $< 5, 3, y >$;
6. fine: $< 0, 0, x > < 0, 0, y > < 2, 1, x > < 3, 2, y > < 5, 3, y >$.

5.3.5: Bit-plane coding

Il bit-plane coding consiste nella composizione multivello dell'immagine in una serie di immagini binarie e comprimere ogni immagine binaria con qualsiasi altra codifica lossless, ottenendo una compressione lossless.

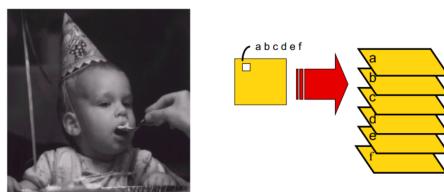


Figura 49: Rappresentazione del bit-plane coding

5.3.6: Standard di compressione lossy

Molto spesso, le immagini vengono compresse in maniera lossy, altrimenti occuperebbero troppo spazio, perciò molto più difficili da trasmettere. Inoltre, tali compressioni non presentano delle perdite significative nella maggior parte degli ambiti in cui si usano, grazie a molti standard. Lo standard più usato e conosciuto è il JPEG, a cui gli viene dedicato l'intero capitolo successivo.

Capitolo 6:

Formato JPEG

Il formato JPEG (Joint Photographic Expert Group) è sicuramente il formato delle immagini più conosciuto ed usato, poiché ottiene dei risultati molto notevoli nelle foto (sia in bianco e nero che a colori), tuttavia non per i cartoni animati e per le immagini generate al computer, nonostante implementi un modello di compressione lossy. In particolare, ne esistono due versioni, entrambe trattate in questo capitolo.

6.1: Prima versione di JPEG

Di seguito è mostrato uno schema che riassume i passi eseguiti nel formato di compressione della prima versione di JPEG.

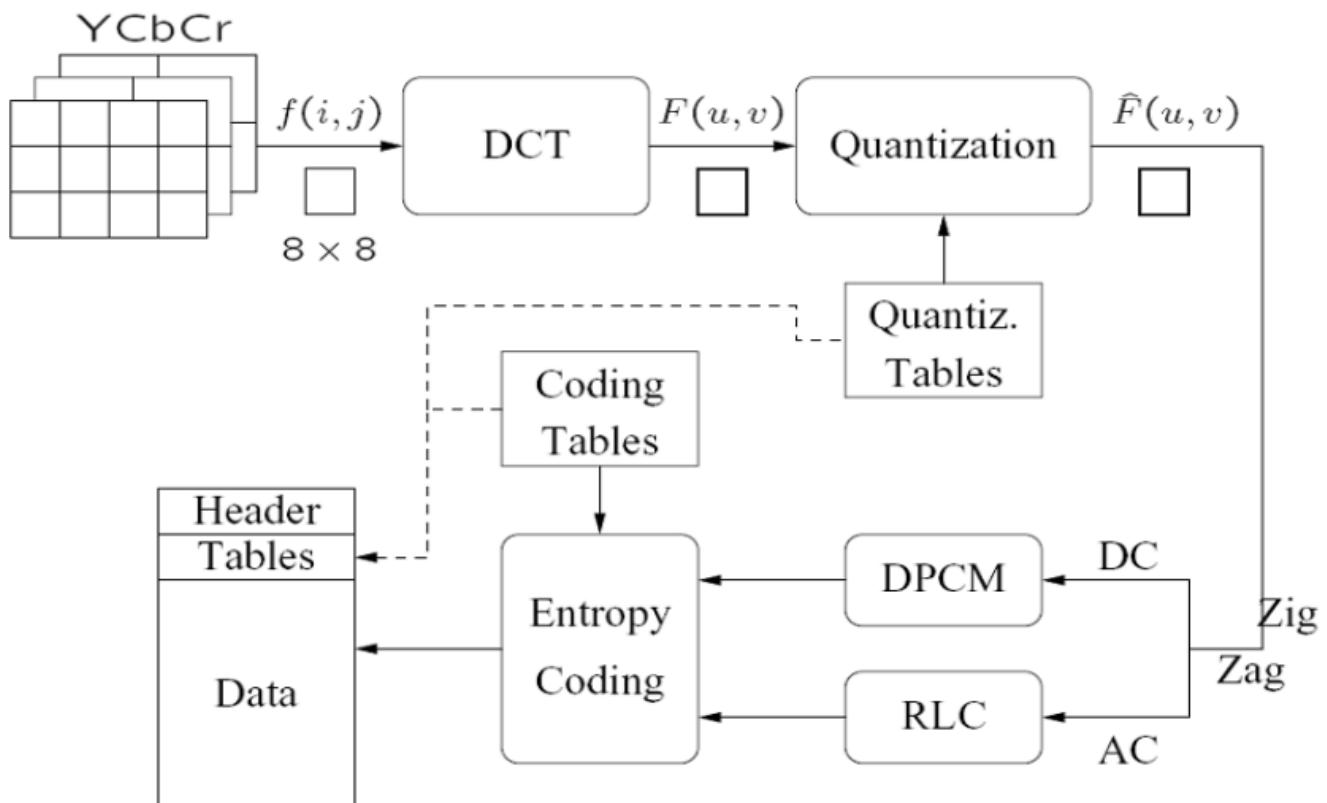


Figura 50: Schema del modello di compressione JPEG

6.1.1: Fase 1: conversione da RGB a YCbCr

Un'immagine a colori viene rappresentata nel formato RGB, che non sono altre tre matrici dove R contiene i livelli di rosso, G di verde e B di blu. tuttavia tale rappresentazione non risulta efficace nella compressione, dato che sono ridondanti. Per questo motivo, si convertono tali matrici nel formato YCbCr, dove Y è luminanza e CbCr rappresentano la crominanza, che insieme non sono affatto ridondanti. Infatti, l'occhio umano percepisce la variazione di luminosità che di colore, dato dal fatto che l'occhio umano ha molti più bastoncelli che coni. Ciò, verrà trattato nei capitoli successivi.

Comunque, la risoluzione delle componenti cromatiche vengono ridotte di un fattore 2. In particolare:

- 4 : 4 : 4 non si ha nessun sottocampionamento;
- 4 : 2 : 2 si ha una riduzione solamente nella direzione orizzontale;
- 4 : 2 : 0 si ha una riduzione sia nella direzione orizzontale sia in quella verticale.

6.1.2: Fase 2: trasformata DCT

A questo punto, viene suddivisa l'immagine in blocchi da 8×8 , poiché è la dimensione che permette la migliore qualità rispetto alla dimensione del file. Per ogni blocco, viene eseguita la cosiddetta DCT (Discrete Cosine Transform), che lavora nell'intervallo da -128 a 128, perciò il blocco deve essere prima traslato negativamente di 128. A questo punto, la formula della DCT è la seguente:

$$G(u, v) = \alpha(u)\alpha(v) \sum_{x=0}^7 \sum_{y=0}^7 g(x, y) \cos\left[\frac{\pi}{8}\left(x + \frac{1}{2}\right)u\right] \cos\left[\frac{\pi}{8}\left(y + \frac{1}{2}\right)v\right]$$

dove la moltiplicazione dei coseni non dipende dai valori di $g(x, y)$ e α è una funzione di normalizzazione. Inoltre, dei 64 coefficienti ottenuti:

- $G(0, 0)$ è il nucleo e viene classificato come coefficiente DC;
- gli altri 63 coefficienti vengono classificati come coefficienti AC.

Infine, i motivi per cui si usa la DCT e non la trasformata di Fourier sono:

- la DCT è reale pura, mentre la trasformata di Fourier è complessa;
- la DCT presenta meno coefficienti di qualsiasi segnale;
- il nucleo della trasformata diretta ed inversa sono gli stessi nella DCT.

6.1.3: Fase 3: quantizzazione

Nella fase di quantizzazione avviene la perdita vera e propria di informazione della compressione. Infatti, viene diviso punto per punto per una determinata matrice e approssimato alla parte intera più piccola. Siccome la quantizzazione avviene in base ad una soglia, viene ridotto il numero di bit per campionamento. La formula è la seguente:

$$T^*(u, v) = \left\lfloor \frac{T(u, v)}{Z(u, v)} \right\rfloor$$

dove:

- $T(u, v)$ è il coefficiente trasformato;
- $Z(u, v)$ è il coefficiente trasformato normalizzato;
- $T^*(u, v)$ è il coefficiente sogliato e quantizzato dell'approssimazione di $T(u, v)$.

6.1.4: Fase 4: pattern zig-zag

A questo punto, vengono ordinati i coefficienti usando un pattern zig-zag, in questo modo vengono ottenute sequenze consecutive di 0 molto più lungh rispetto che farlo per riga.

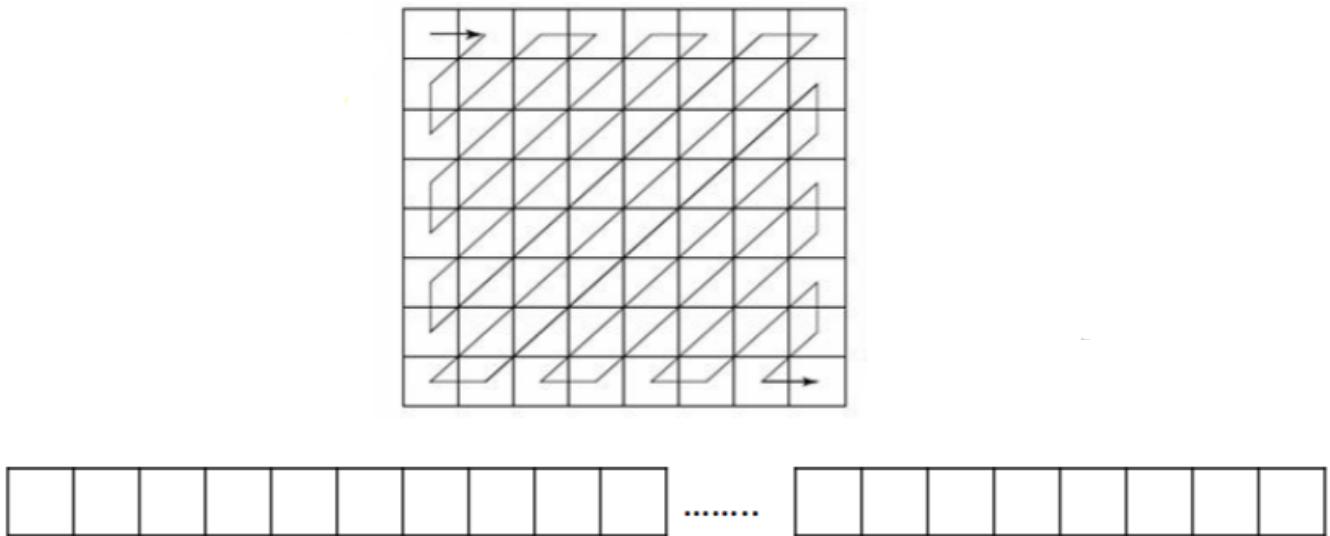


Figura 51: Pattern zig-zag

6.1.5: Fase 5: Codifica di entropia

A questo punto:

- tutti i coefficienti DC vengono codificati con la DPCM (Differential pulse-code modulation), che non è altro che la differenza tra i coefficienti DC ed i coefficienti DC dell'immagine precedente;
- tutti i coefficienti AC vengono codificati utilizzando la RLC, siccome sono presenti molti 0 consecutivi.

Infine, tutti i coefficienti vengono codificati in una sequenza binaria, come quella di Huffmann e quella aritmetica; ed infine viene rieseguita la IDCT per riottenere l'immagine compressa.

6.2: JPEG2000

La seconda versione di JPEG, denominata JPEG2000, è una versione molto migliorata rispetto alla precedente, sebbene purtroppo sia ad oggi poco utilizzata nell'ambito comune, per la scarsa implementazione nei browser principali. Tuttavia, trova largo utilizzo in ambiti più tecnici, come nella medicina, dove è necessaria una risoluzione molto più alta.

Questo standard presenta:

- migliori performance di velocità/distorsione;
- possibilità di definire delle regioni di interesse (ROI, dall'inglese regions of interest), per avere maggiore qualità;
- funzionamento anche con immagini veramente molto grandi.

Tuttavia, il vantaggio più importante di JPEG2000 è sicuramente la scalabilità dei seguenti quattro tipi:

- scalabilità di risoluzione, ossia è possibile ottenere un flusso progressivo di immagini, partendo dalla risoluzione più bassa;
- scalabilità di distorsione, cioè è possibile ottenere un flusso progressivo di SNR basso;
- scalabilità spazioale, ovvero è possibile ottenere un flusso progressivo di una specifica regione, partendo da una piccola ridotta;
- scalabilità dei componenti, dove possibile ottenere un flusso progressivo, partendo da una scala di grigi per arrivare ad un'immagine a colori.

Di seguito, è elencato uno schema riassuntivo del processo di compressione di JPEG2000.

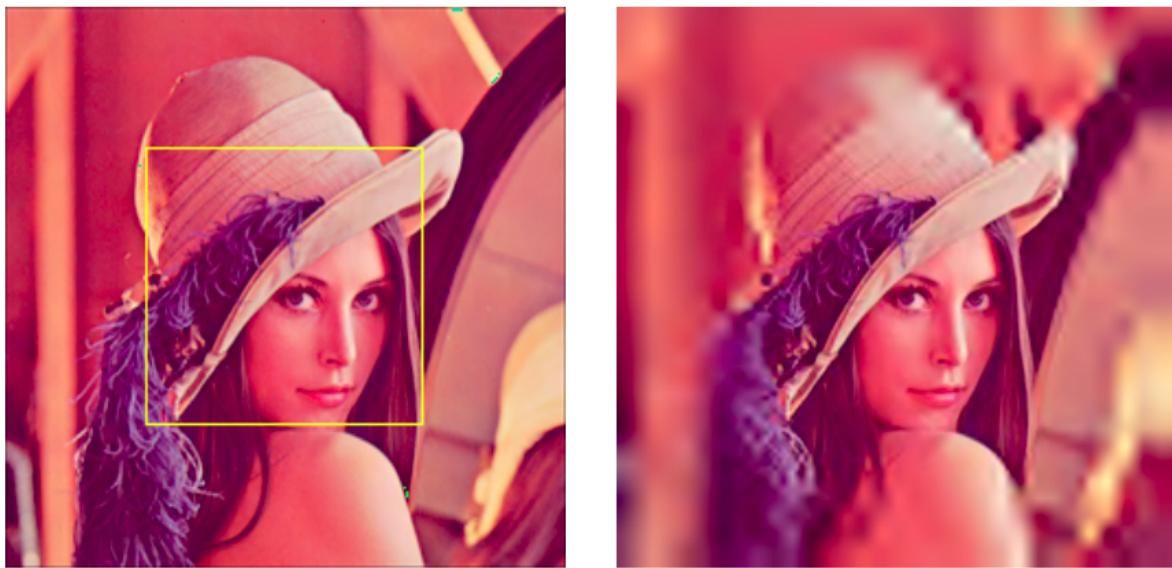


Figura 52: Esempio di applicazione ROI dall’immagine originale (a sinistra) all’immagine compressa (a destra)

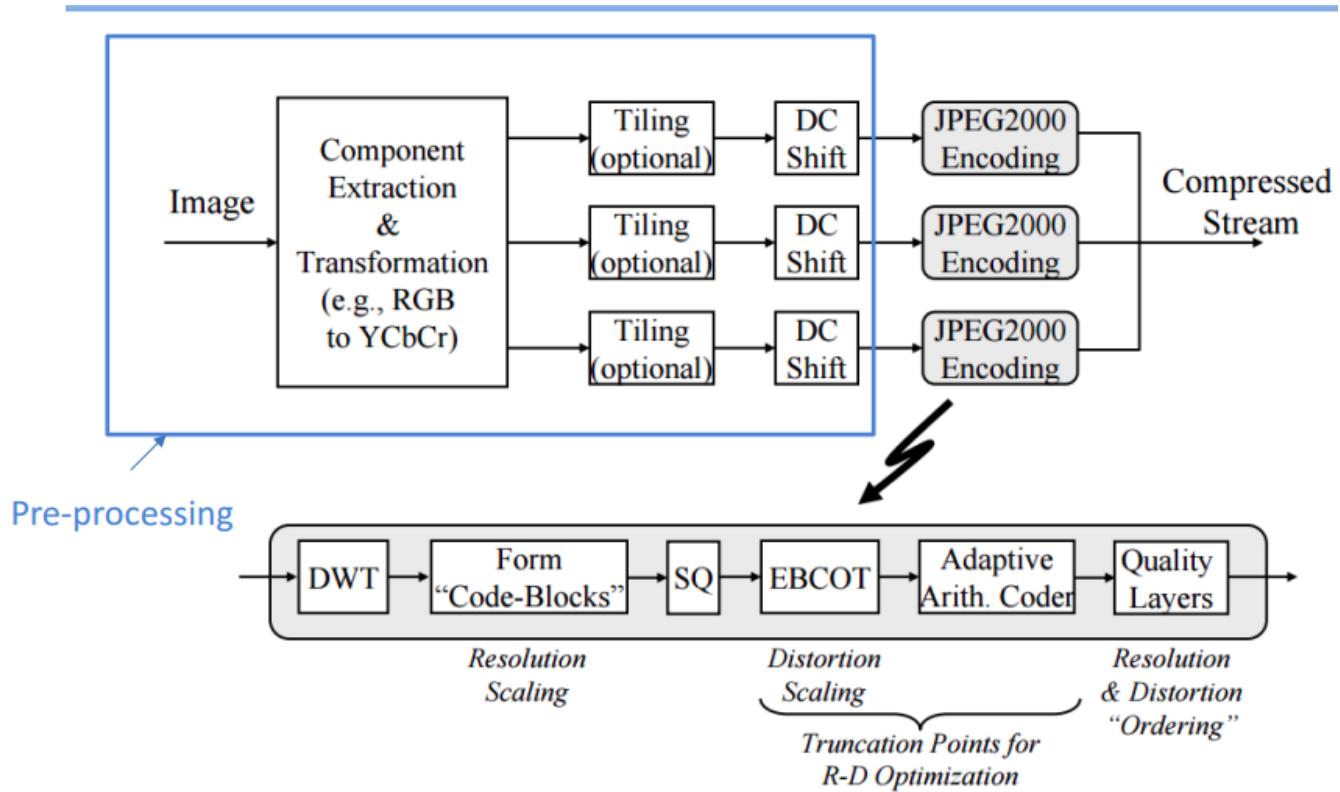


Figura 53: Schema del modello di compressione JPEG2000

6.2.1: Fase 1: preprocessing

Per prima cosa, si esegue il cosiddetto preprocessing, che si compone in tre sottofasi, di cui alcune sono molto simili alla prima versione di JPEG.

Innanzitutto, si effettua l'estrazione delle componenti e la trasformazione, in cui si mappano le componenti RGB e si trasformano in YCbCr, per ridurre la correlazione tra le componenti, conducendo a due tipologie di trasformazioni:

- trasformazione delle componenti irreversibile (ICT, irreversible component transformation);
- trasformazione delle componenti reversibile (RCT, reversible component transformation).

Successivamente, si esegue la cosiddetta piastrellatura dell'immagine (in inglese image tiling), in cui avviene una partizione dell'immagine in blocchi rettangolari non sovrapposti, in modo tale da poter essere compressi in maniera indipendente.

Infine, avviene la traslazione dei coefficienti di -128 , passando dall'intervallo $[0, 255]$ a $[-128, 128]$, per avere la trasformata centrata in 0.

6.2.2: Fase 2: discrete Wavelet transform

A questo punto, ad ogni blocco viene effettuata la DWT (Discrete Wavelet Transform), poiché:

- permette la decomposizione multirisoluzione;
- ammette la scalatura di risoluzione;
- ogni livello di bassa risoluzione, viene suddiviso in sottobande di dimensioni dimezzate.

In particolare, JPEG2000 supporta da 0 a 32 stadi: solitamente per le immagini naturali, se ne usano tra 4 ed 8.

6.2.3: Fase 3: codifica di blocchi

Nella fase di codifica dei blocchi, ogni sottobanda viene partizionata in codifiche di blocchi, tipicamente di dimensione 32×32 o 64×64 , in cui ogni blocco viene codificato separatamente. In questo modo, si ha la scalatura di risoluzione e spaziale.

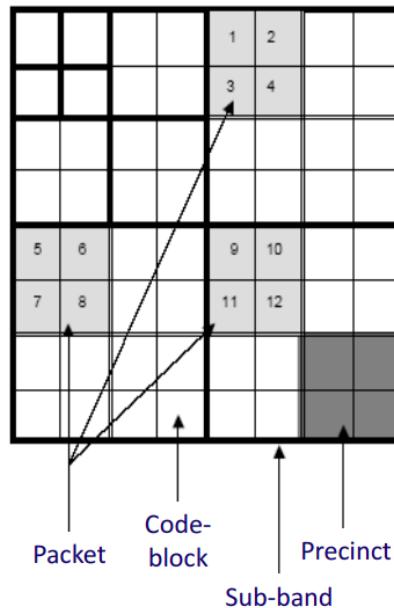


Figura 54: Schema della fase di codifica di blocchi

6.2.4: Fase 4: quantizzazione

Nella fase di quantizzazione, i coefficienti wavelet sono quantizzati usando un quantizzatore uniforme, in cui per ogni sottobanda b e la larghezza del gradino del quantizzatore di base Δ_b , si quantizza ogni coefficiente, sfruttando la formula seguente:

$$q = \text{sign}(y) \left\lfloor \frac{|y|}{\Delta_b} \right\rfloor$$

Di seguito, è riportato un esempio con $y = -21,7$ e $\Delta_b = 10$.

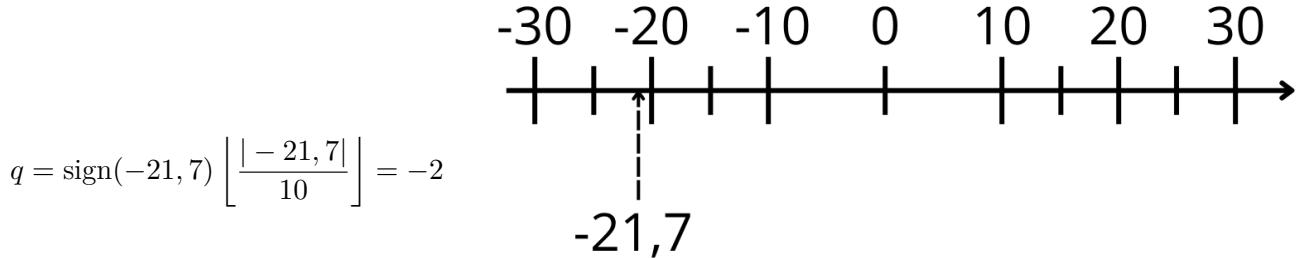


Figura 55: Esempio di quantizzazione JPEG2000

6.2.5: Fase 5: EBCOT

La fase di EBCOT (Embedded Block Coding with Optimized Truncation) consiste nel prendere un blocco e codificarlo con il bit-plane dal bit più significativo a quello meno significativo. Inoltre, se alcuni piani più significativi non contengono 1, allora il piano viene impostato sul piano di bit più in alto, con almeno un 1. In questo modo, si ha una classificazione dei coefficienti, partendo da quelli più insignificanti, classificati come 0, finché non viene codificato il primo non-zero, diventando significativo, codificando il suo segno e tutta la sua sottosequenza di bit. Ad esempio, i dati con la riduzione di distorsione più alta dovrebbero essere codificati per primi per media di rappresentazione di bit compressi. Per essere codificato, il coefficiente deve superare:

- la propagazione di significatività, in cui se è un bit è insignificante, ma almeno uno dei suoi vicini lo è, allora viene codificato, oppure allora stesso tempo è significante, e il suo flag è 1, allora il segno del simbolo è codificato;
- la raffinatezza di magnitudine, che codifica campioni significativi che non sono stati codificati nel passaggio precedente;
- il passaggio di pulizia, in cui codifica tutti i bit che sono stati passati attraverso i due passaggi di codifica precedenti.

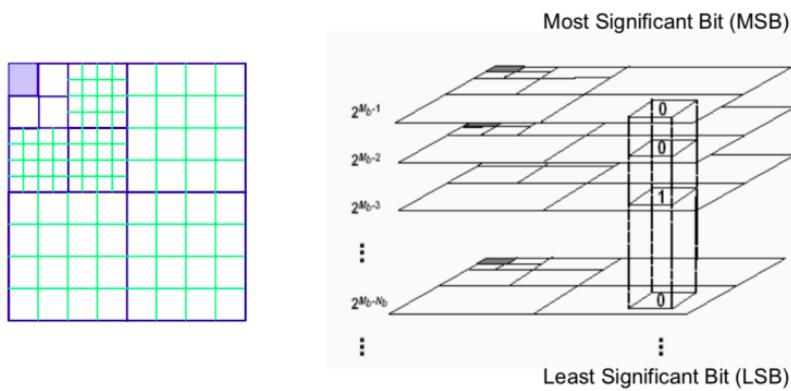


Figura 56: Fase di EBCOT

6.2.6: Fase 6: codifica aritmetica adattiva

Ora, per ridurre la ridondanza sostanziale tra piani di bit successivi, si usa la codifica aritmetica adattiva, in cui è in grado di cambiare tra ben 18 modelli di probabilità adattiva. In particolare:

- seleziona i modelli basati sulla bit codificati precedentemente dal piano corrente a quelli precedenti;
- ogni modello stima la sua distribuzione di probabilità;
- in questo contesto usa la probabilità condizionale.

6.2.7: Fase 7: strati di qualità

I flussi di bit risultanti per ogni blocco di codice sono organizzati in livelli di qualità per ottenere:

- scalabilità della risoluzione, cioè l'eliminazione dei componenti dalle sottobande ad alta risoluzione;
- scalabilità della distorsione, ossia l'eliminazione degli bit meno significativi dalla quantizzazione incorporata;

I livelli di qualità consentono la gestione di questa struttura di scalabilità:

- specificando come ogni blocco deve essere troncato rispetto agli altri;
- fornendo ad ogni livello nel flusso incorporato un miglioramento progressivo;
- ottimizzando la struttura dei livelli durante la compressione, per un embedding ottimale.

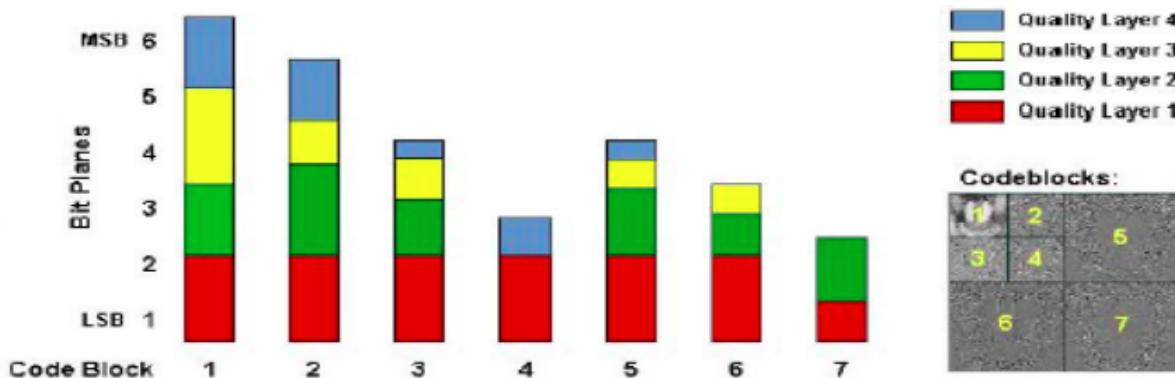


Figura 57: Strati di qualità

Capitolo 7:

Fondamenti della percezione della profondità

In questo capitolo, viene trattato come avviene la percezione dello spazio tridimensionale, partendo solamente da un'immagine, che invece possiede solamente due dimensioni. Per prima cosa, è necessario comprendere il funzionamento del nervo oculomotore. Per prima cosa, avviene l'accomodamento, in cui il cristallino si comporta come una lente convergente di distanza focale variabile. Inoltre, il muscolo ciliare si contrae e si allunga, variando forma e quindi la distanza focale. Perciò, nel momento in cui si guardano oggetti lontani, il cristallino si appiattisce, mentre quando si osservano oggetti vicini diventa più spesso. In



Figura 58: Processo di adattamento

contemporanea, avviene la cosiddetta vergenza, che non è altro che un movimento simultaneo degli occhi in direzioni opposte, che consentono la fusione. In particolare:

- gli occhi ruotano uno verso l'alto per guardare oggetti vicini, detta convergenza;
- gli occhi ruotano in direzioni opposte per guardare oggetti lontani, denominata divergenza.

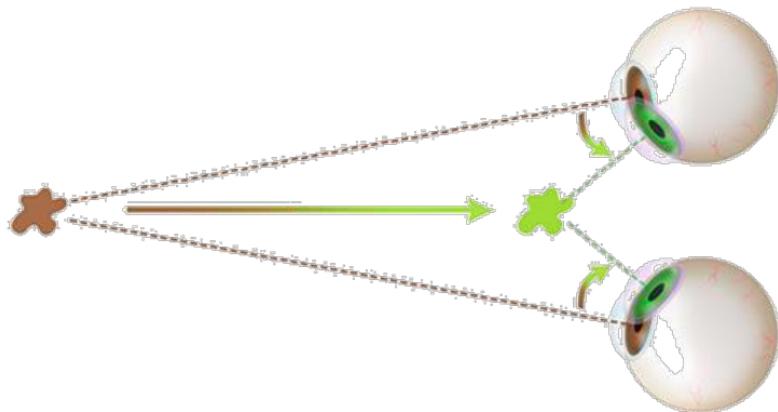


Figura 59: Processo di vergenza

7.1: Indicatori monoculari di profondità statici

In questo paragrafo, vengono trattati i cosiddetti indicatori monoculari di profondità statici, cioè quelle caratteristiche che forniscono delle informazioni tridimensionali in immagini. Come schema, conviene analizzare il dipinto *A Rainy Day in Paris*.



Figura 60: *A Rainy Day in Paris* che mostra le occlusioni (1), le dimensioni relative (2), le tessiture (3), la prospettiva lineare (4), la prospettiva aerea (5) e le ombre (6)

7.1.1: Occlusioni

Il primo indicatore sono sicuramente le occlusioni, che si definiscono come gli oggetti più vicini che bloccano l'accesso visivo a quello più distante. Per esempio, nella figura seguente viene mostrata una ragazza che viene parzialmente coperta da un tronco di una palma.



Figura 61: Esempio di occlusione

7.1.2: Dimensioni relative

Le dimensioni relative forniscono informazioni 3D, grazie ad oggetti della stessa dimensione fisica proiettano immagini retiniche di dimensione diversa a seconda del punto del punto di osservazione. Per esempio, è molto più facile conoscere l'altezza di una statua dell'isola di Pasqua con delle persone (che hanno un'altezza media di 1,80 m), rispetto a che non ci sono persone od altri oggetti di riferimento: infatti, anche l'esperienza gioca un ruolo fondamentale in questo contesto.



Figura 62: Esempio di dimensioni relative

PARTE II:

VIDEO

Capitolo 8:

Creazione dei video