

Enhancement in the Spazial & Frequency Domain

Ci sono due tipi di enhancement nel campo della elaborazione delle immagini:

- Miglioramento nel Dominio dello Spazio
- Miglioramento nel Dominio della Frequenza

La prima, quindi, fa modifiche su i pixel e sulla intensità che il sensore pixel ha registrato mentre la seconda fa modifiche e elaborazioni sui Coefficienti di una trasformata matematica.

Dominio Spaziale:

Tratteremo prima del Dominio dello Spazio → dove preso un intensità del pixel si ha una funzione che modifica quell'intensità.

Questa prima tecnica è chiamata Point Operators/Intensity Transformations:

$$I(x, y) \rightarrow J(x, y)$$

dove il valore finale dipende dal valore iniziale passata alla trasformazione.

Esiste anche un'altra operazione che coinvolge un gruppo di pixel e si chiama Spatial Operations o Spatial Filtering. Questa operazione, ritorna un risultato che è influenzato anche da i pixel vicini al pixel in ingresso.

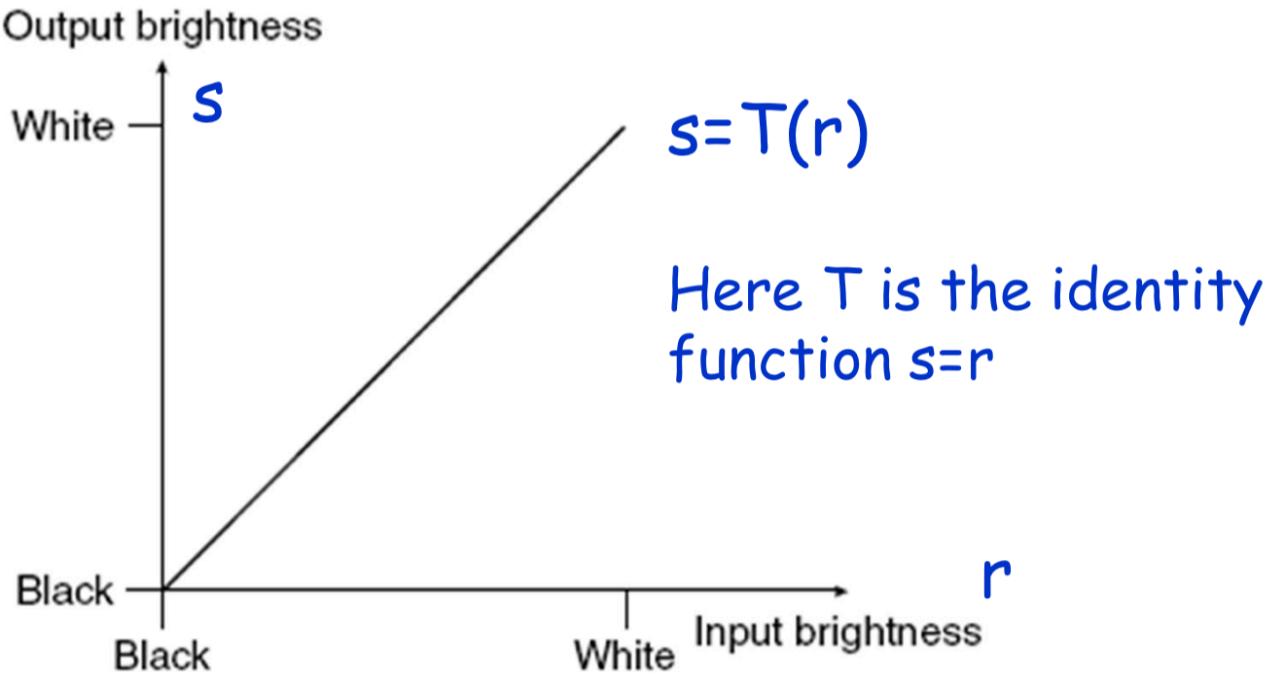
Point Operations

Queste operazioni processano un intensità registrata da un pixel in un'altra come se fosse una funzione. $s = T(r)$ dove r è il pixel originale. L'output in una locazione è dipendente da il valore dell' input (che è nella stessa locazione).

Differenza tra **Brightness** e **Contrasto**:

- **Brightness:** si riferisce all'intera chiarezza/ scurezza dell'immagine.
- Contrast: si riferisce alla differenza tra zone illuminate della foto e zone scure, che rendono gli oggetti più distinguibili.

Tornando quindi alla definizione di queste operazioni, poichè questa operazione può essere vista come una trasformazione da un valore all' altro. Allora si può ricondurre ad una funzione e mappare i valori di input e di output di un intensità di un pixel.



(Nelle immagini RGB vengono mappate tre bande(Red, Green, Blue) quindi la trasformazione sarà applicata tra tre valori di pixel $J(r, c, b) = T_b(r, c, b) \forall b = 1, 2, 3$ per tutti (m, n)).

Operazioni Lineari

Vediamo quindi che una trasformazione lineare può essere trasformata così:

$$T(r) = a \cdot r + g$$

- g : è il parametro che controlla la luminosità.
- a : è il parametro che controlla il contrasto.

Vediamo un pò di trasformazioni.

Aumento Luminosità:

$$J(r, c, b) = \begin{cases} I(r, c, b) + g & \text{se } I(r, c, b) + g < 255 \\ 255 & \text{se } I(r, c, b) + g \geq 255 \end{cases}$$

diminuire la luminosità è la stessa cosa ma con $g < 0$.

Aumentare il Contrasto:

La trasformazione è $T(r, c, b)$ mentre il risultato finale ottenuto è rappresentata da $J(r, c, b)$,

$$T(r, c, b) = a \cdot [I(r, c, b) - z] + z$$

$$J(r, c, b) = \begin{cases} 0 & \text{se } T(r, c, b) < 0 \\ T(r, c, b) & \text{se } 0 \leq T(r, c, b) \leq 255 \\ 255 & \text{se } T(r, c, b) \geq 255 \end{cases}$$

$a > 1 \quad z \in 0, \dots, 255 \quad b \in 1, 2, 3$

mentre per la diminuzione del contrasto a deve essere compreso tra 0 e 1.

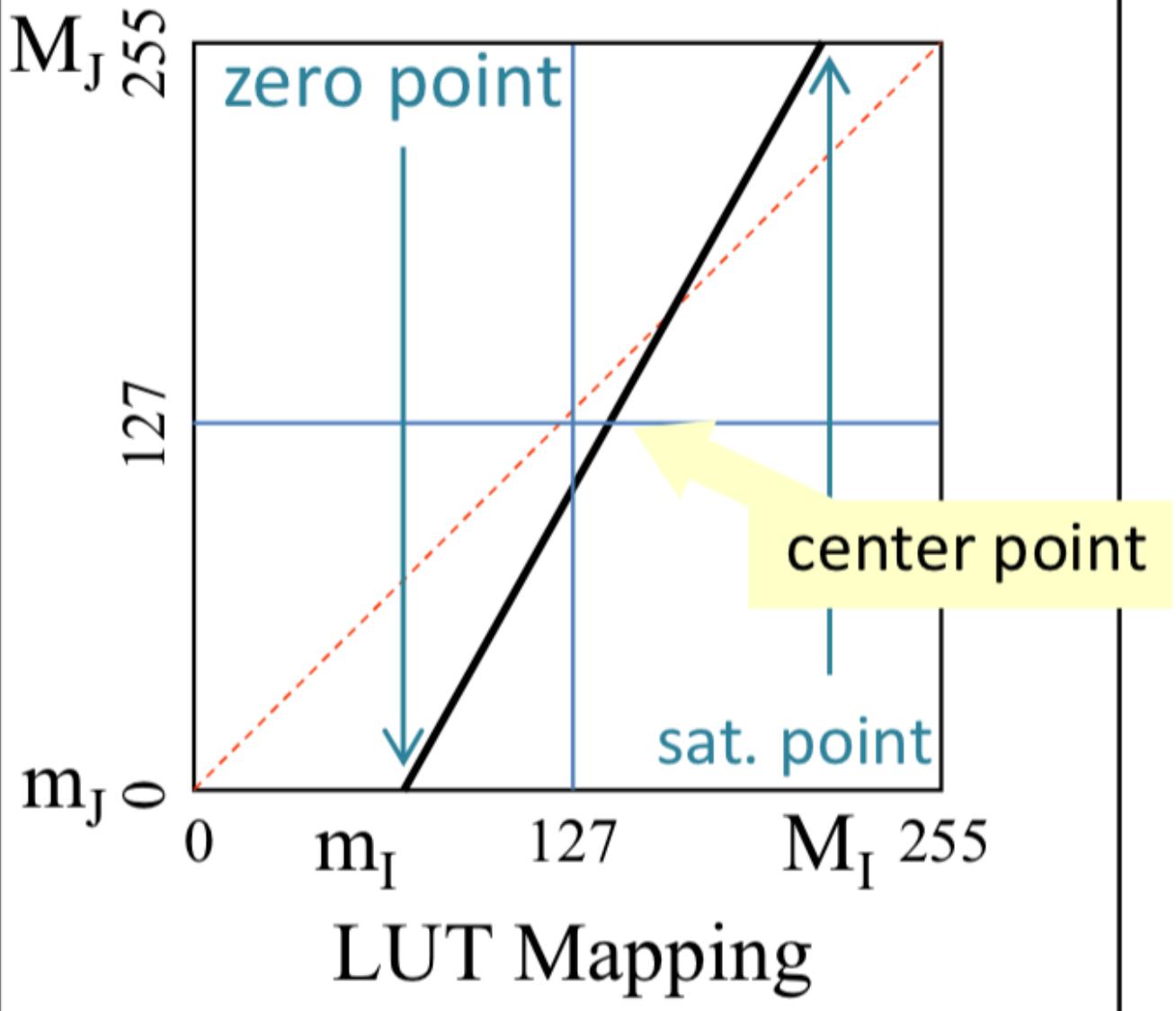
Stretch del Contrasto

Let $m_I = \min[\mathbf{I}(r, c)]$, $M_I = \max[\mathbf{I}(r, c)]$,
 $m_J = \min[\mathbf{J}(r, c)]$, $M_J = \max[\mathbf{J}(r, c)]$.

Then,

$$\mathbf{J}(r, c) = (M_J - m_J) \frac{\mathbf{I}(r, c) - m_I}{M_I - m_I} + m_J.$$

Notiamo che M_J e m_J sono valori prestabiliti, mentre gli altri m_I e M_I sono i valori max e min dei pixel nell'immagine.



Zero point: indica il valore di output corrispondente al valore minimo di input (m_I).

Saturation point: indica il valore massimo (M_J) mappato all'intensità massima M_I dell'immagine originale.

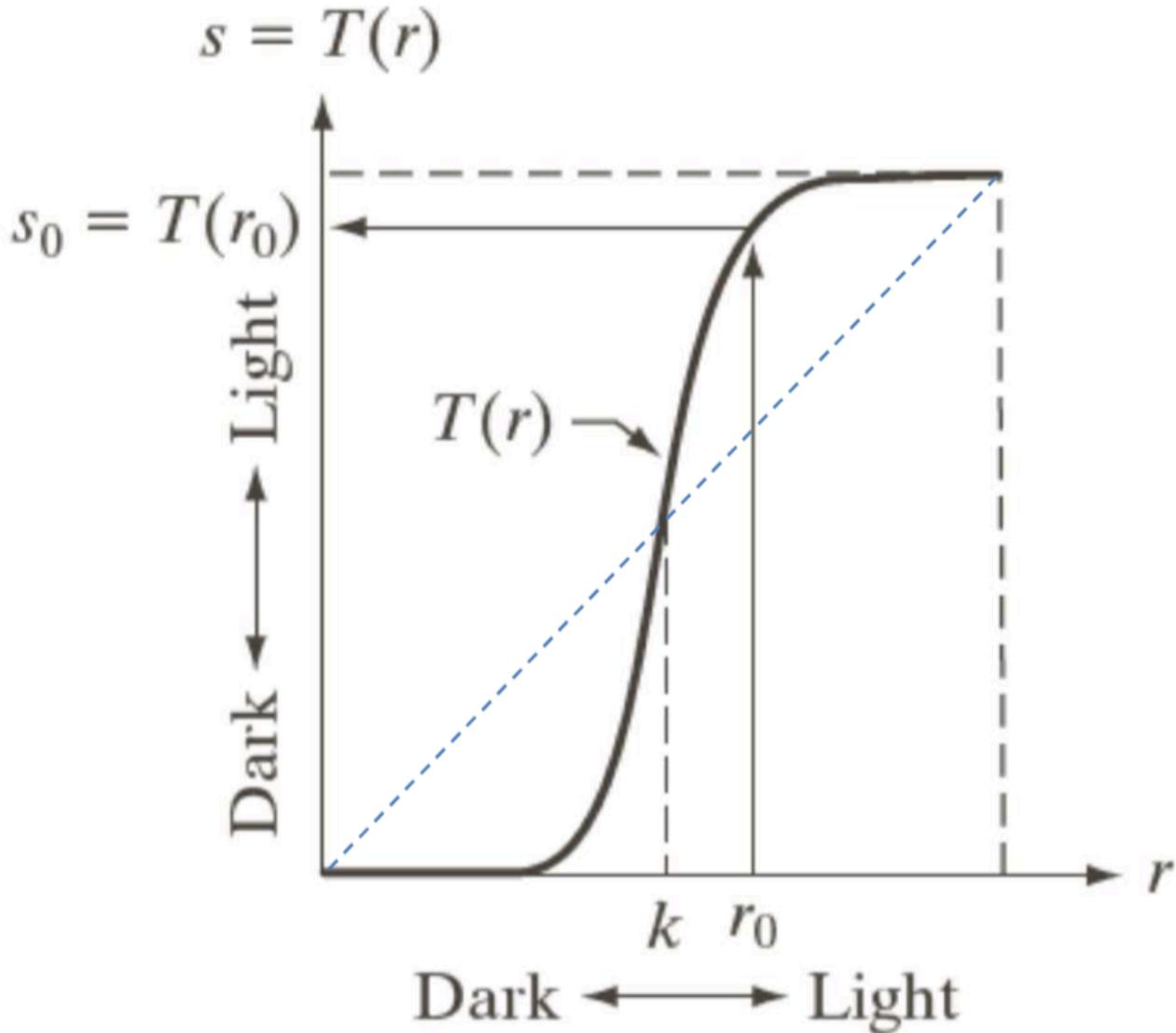
Importante è anche l'autocontrasto o anche *aggiustamento automatico del contrasto* che mappa il pixel più scuro dell'immagine in input come 0 e il più luminoso come 255 (se $L=256$) e ridistribuisce i valori intermedi usando questa equazione:

$$s = \frac{L - 1}{r_{max} - r_{min}} \cdot (r - r_{min})$$

Operazioni Non Lineari

Stretch del contrasto

Se $T(r)$ ha una forma particolare tipo questa:



Vediamo che sotto k i valori dei pixel iniziano a scurirsi mentre sopra k iniziano a schiarirsi. La funzione è rappresentata così:

$$s = T(r) = \frac{1}{1 + (k/r)^E}$$

dove maggiore è la E e maggiore si avvicina ad un comportamento verticale (controlla la pendenza). Se k varia si sposta il limite lungo r ...

Se la pendenza è molto alta allora si ha un altro tipo di operazione che è chiamata **Thresholding** dove il pixel luminoso sopra k diventa bianco e il viceversa diventa nero.

Trasformazioni da applicare sui livelli di grigio

Immagini Negative

Le immagini negative vengono fatte attraverso questa formula: $s = T(r) = (L - 1) - r$. Essa serve per far apparire i pixel luminosi scuri e viceversa.

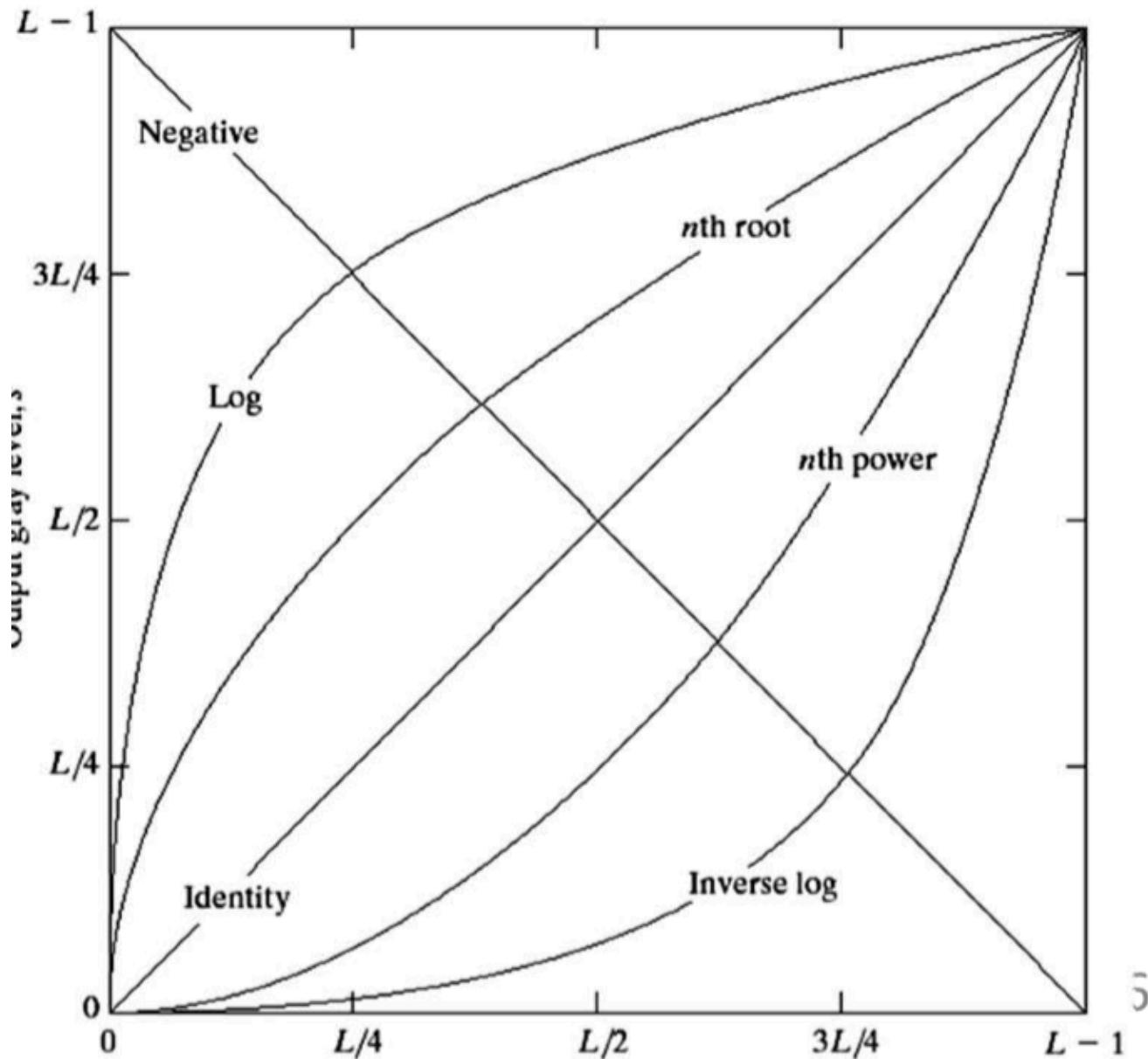
Serve a mettere in evidenza dei dettagli.

Trasformazioni Logaritmiche

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

dove vediamo che quindi parte da origine e l'intervallo di r è comunque da 0 a $L-1$.

Se vediamo la figura di questa trasformazione:



Si nota che, i pixel più scuri in input hanno un più largo intervallo di mappatura rispetto ai pixel con valori più alti. Mentre l'effetto opposto è dato dalla trasformazione inversa, che in questo caso è l'esponenziale.

Si utilizza la trasformazione logaritmica quando i livelli di grigio in ingresso hanno un largo intervallo di valori. Questa operazione fa sì che si comprimi la *image dynamic range*.

Esempio:

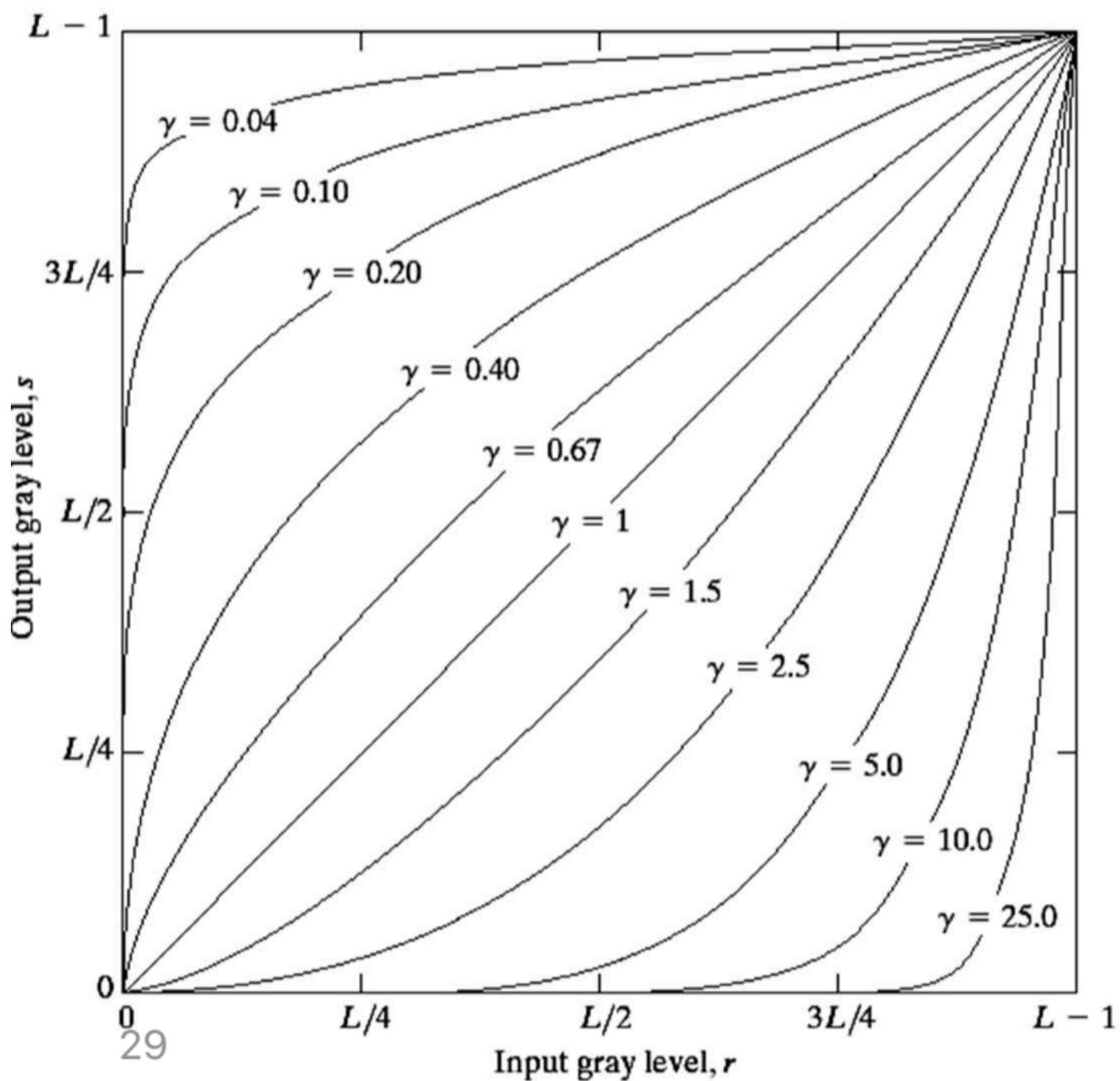
i display systems non riproducono una grande varietà di valori di intensità dei pixels, di conseguenza ci saranno perdite nello spettro di Fourier.

Quindi se i valori stanno da 0 a 10^6 se si attua una compressione arrivano $\rightarrow 0$ a 6

Legge di Potenza

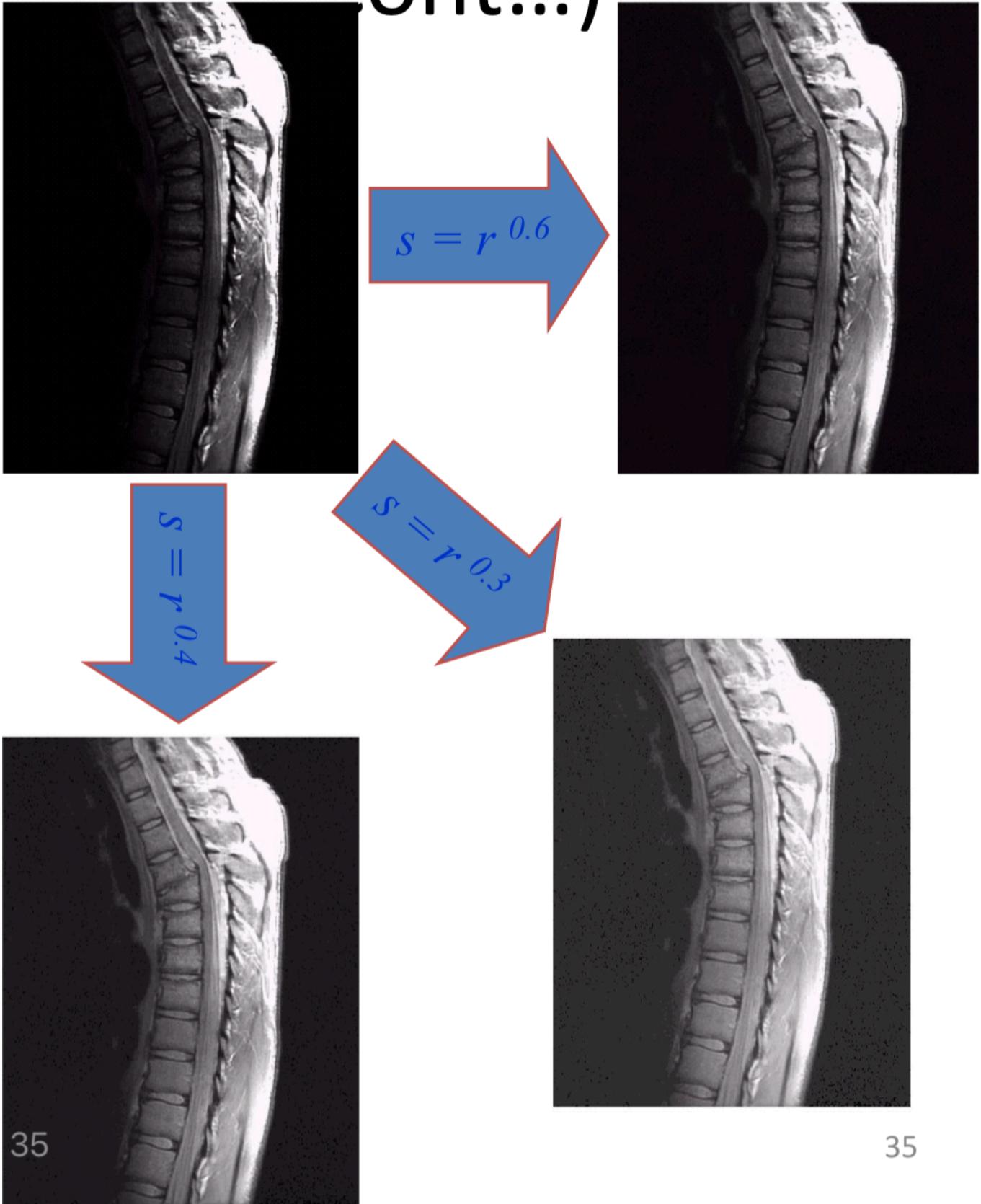
L'equazione di questa trasformazione è: $s = c \cdot r^\gamma$

Vediamo il grafico della trasformazione:



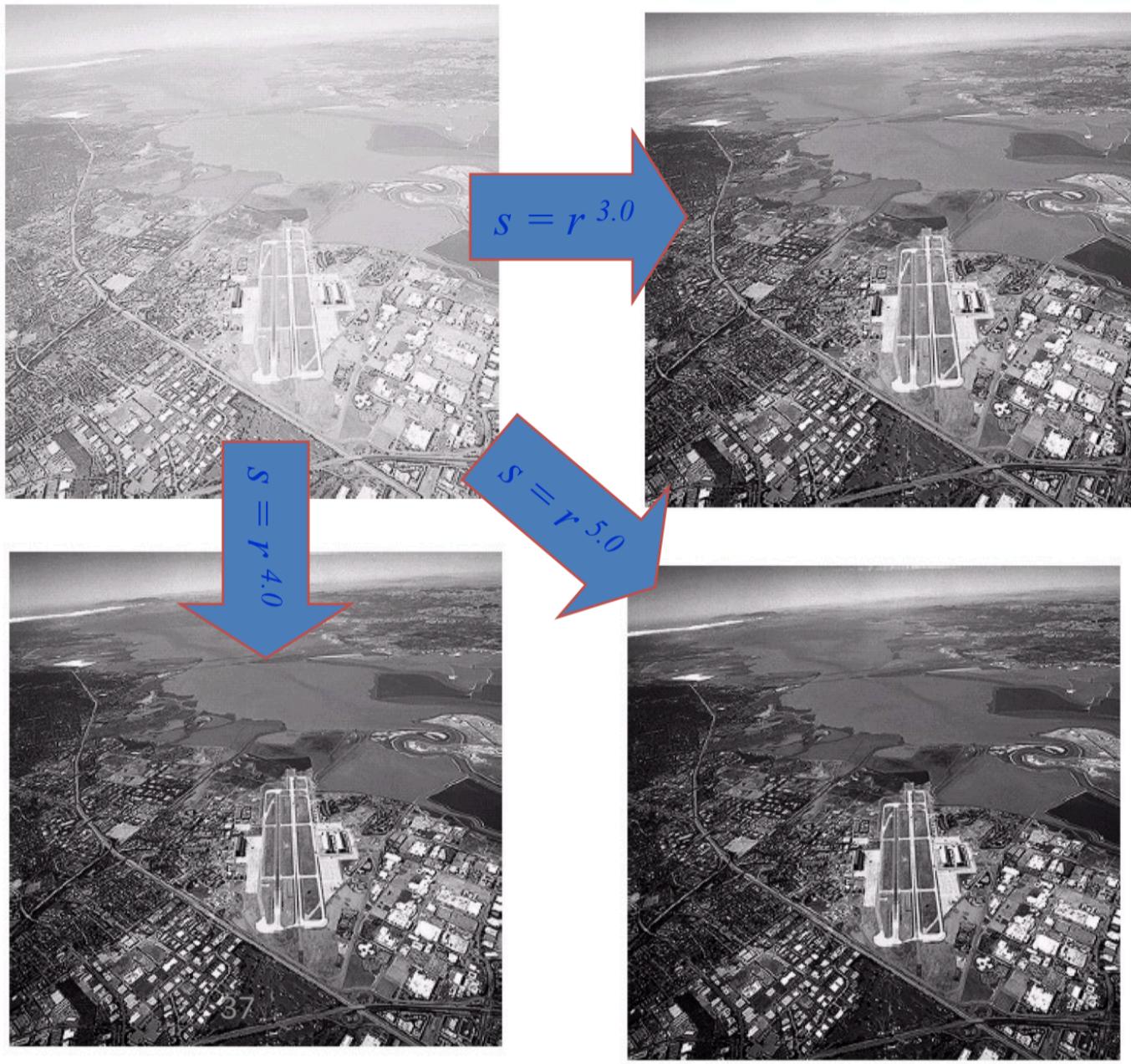
Vediamo che con $\gamma < 1$ si espande i valori scuri in valori di uscita più chiari, mentre si ha il viceversa per $\gamma > 1$.

Al contrario del logaritmo la potenza crea a sua volta dentro questa distinzione da gamma altre famiglie di curve che possono essere ottenute. Vediamo nel caso di una radiografia magnetica come si può migliorare la foto facendo un aumento del livello di grigi attraverso una trasformazione con Potenza e $\gamma < 1$



Da questa foto si può vedere che la migliore trasformazione è attraverso la trasformazione con $\gamma = 0.4$, infatti utilizzando gamma minori la foto diventa lavata, perdendo qualità e dettagli.

Mentre nel caso gamma sia maggiore di 1, si può utilizzarli riducendo la banda dei valori più chiari in uscita.

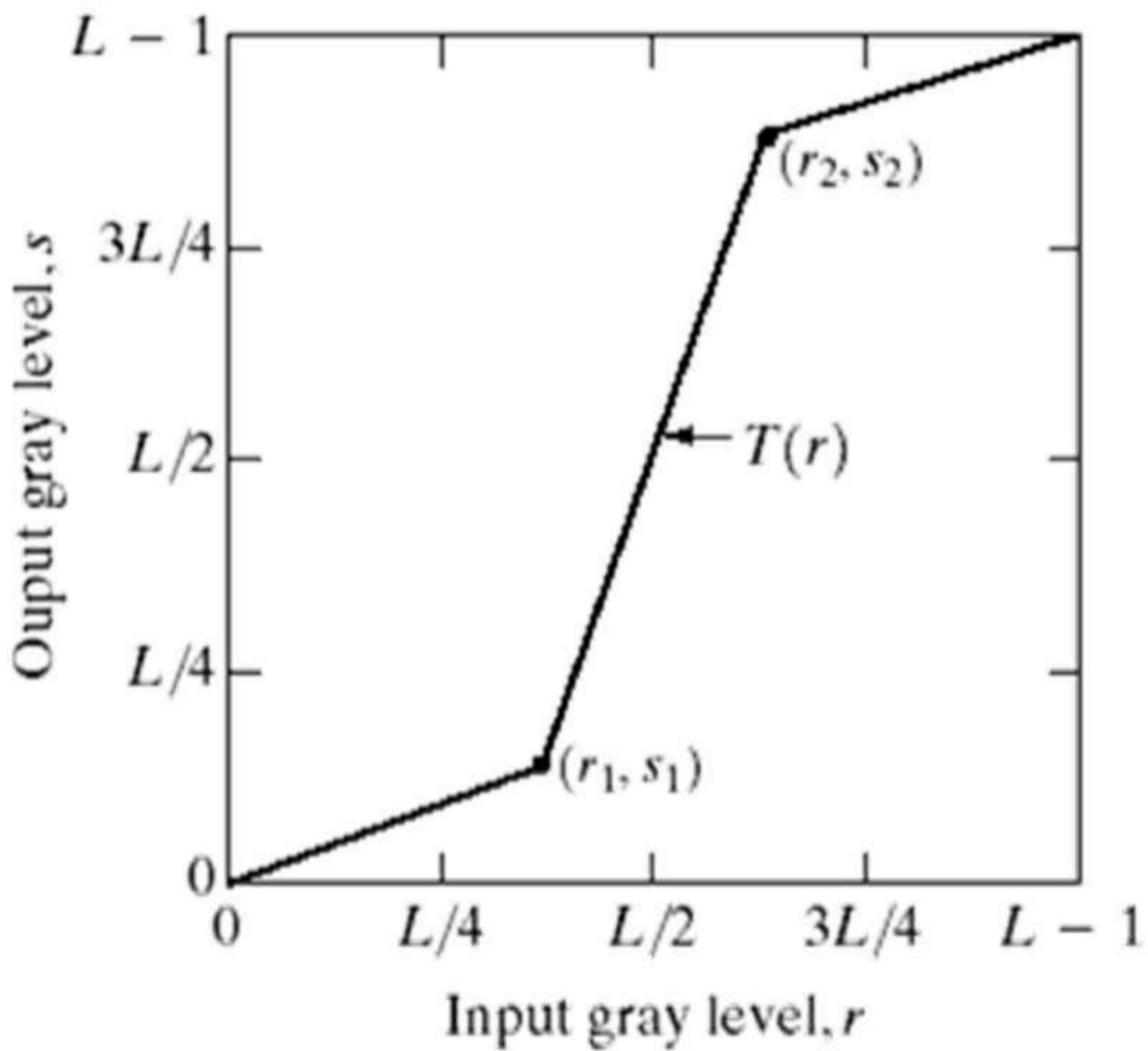


In questo gruppo vediamo che si acquista più dettagli nei livelli di grigio se si adotta $\gamma = 4.0$ poichè con 5.0 si perdono dettagli nelle zone più scure.

Stretching del Contrasto & Thresholding

Sostanzialmente questa trasformazione ci permette di avere più gradi di libertà poichè è una trasformazione formata da segmenti.

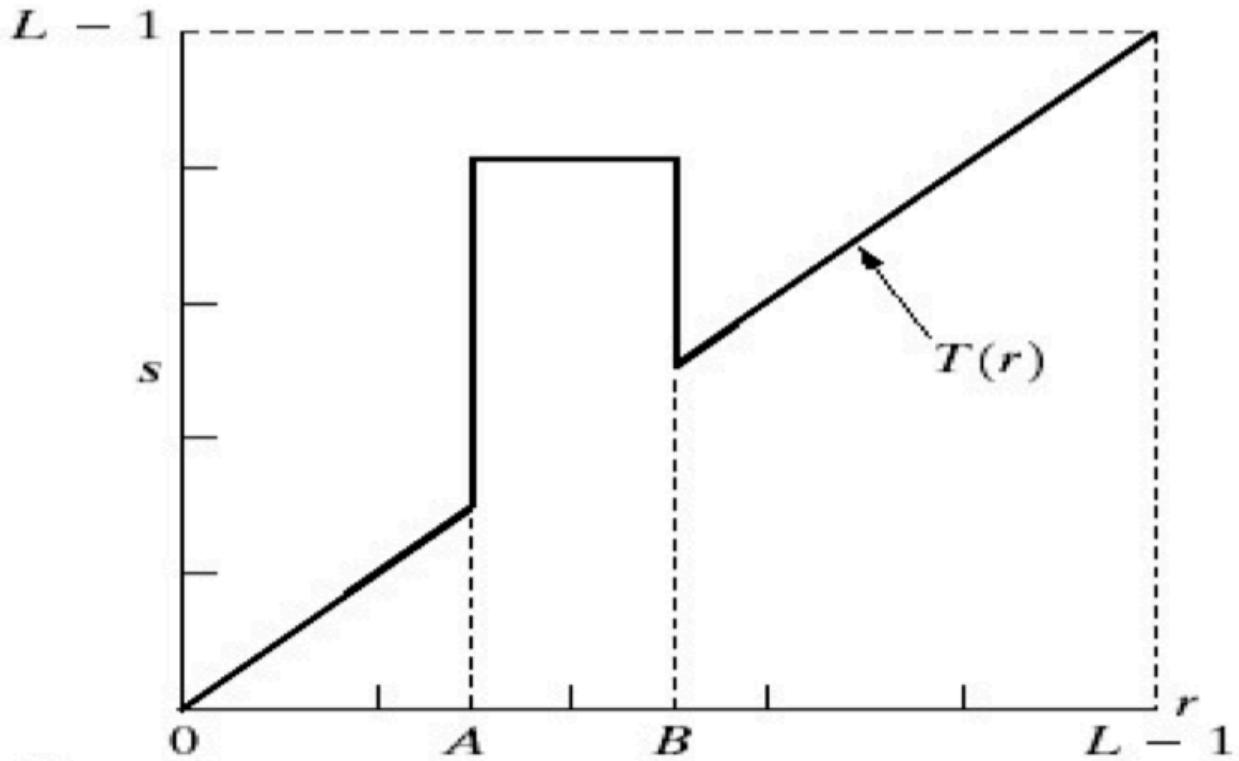
Come vediamo dalla figura.



La figura fa capire in che senso si parla di gradi di libertà, infatti se posizioniamo $r_1 = r_2$ otteniamo un thresholding della figura e variando quindi (r_1, s_1) e (r_2, s_2) otteniamo vari output.

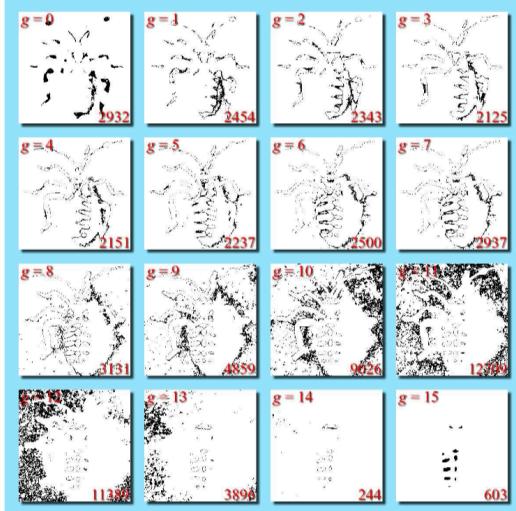
Intensity Level Slicing

Questa operazione fa sì che solo un intervallo di livelli di grigio venga modificato, gli altri livelli rimangono lo stesso.

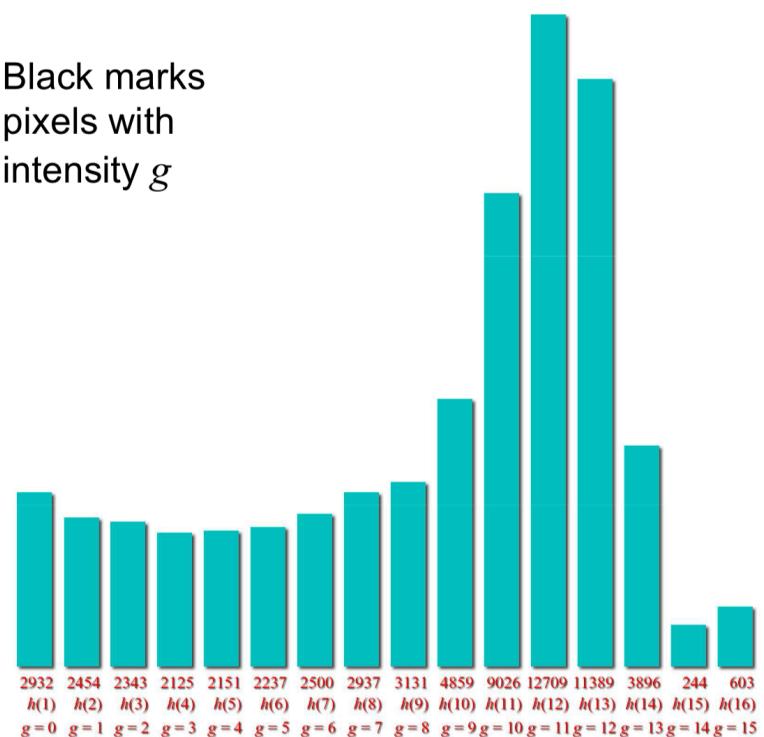


Istogramma dell'immagine

Questi istogrammi rappresentano il numero di pixel che hanno registrato un eventuale valore. Infatti, mostra a noi la distribuzione dei livelli di intensità dando anche una stima dell'**occorrenza dei livelli**. Servono per tecniche di processamento spaziale.



Black marks
pixels with
intensity g



Plot of histogram:
number of pixels with intensity g

Nelle colonne ci sono il numero di pixel con intensità g .

Quindi vediamo che in questo caso il livello di grigi è da 0 a 15, quindi sono 16 livelli dove:

- g_k : è il k^{th} livello di grigio.
- n_k : è il numero di pixel che ha registrato quel tono di grigio.

Histogram Equalization

Più l'istogramma è uniforme e più l'immagine è contrastata bene.

Di conseguenza vogliamo cercare una funzione che dato l'istogramma, si uniformi tutto.

Ricordiamo che l'istogramma ha sull'asse delle x i valori che possono assumere i pixel mentre sulla y la frequenza di registrazione del valore.

Vogliamo, quindi, una funzione che prende in input r (valore di pixel compreso tra 0 e L-1) e restituisce un altro valore.

Le ipotesi di questa funzione sono:

- T deve essere a valore singolo (per avere inversa) e monotona crescente tra $0 \leq r \leq 1$ (normalizzato 1 è 255 0 è 0). (La monotonia crescente è data dal fatto che se in entrata entra un valore nero allora ne deve uscire uno nero)
- $0 \leq T(r) \leq 1$ per $0 \leq r \leq 1$

In un'immagine, però, i livelli di grigio possono essere visti come randomici. Quindi si può vedere una densità di probabilità del valore r .

Sappiamo dalla teoria della probabilità che se $p(r)$ e $T(r)$ sono conosciute e soddisfa la prima ipotesi allora:

$$p_s(s) = p_r(r) \cdot \left| \frac{dr}{ds} \right|$$

Quindi se scegliamo $T(r)$ come una funzione di distribuzione cumulativa:

$$s = T(r) = \int_0^r p_r(w) dw$$

Allora vediamo che:

$$\frac{ds}{dr} = \frac{dT(r)}{dr} = \frac{\int_0^r p_r(w) dw}{dr} = p_r(r)$$

quindi viene:

$$p_s(s) = p_r(r) \left| \frac{1}{p_r(r)} \right| = 1$$

Quindi la seconda ipotesi è verificata.

Con questa funzione, sappiamo che $T(r)$ dipende da $p_r(r)$ ma $p_s(s)$ è sempre uniforme indipendentemente da $p_r(r)$.

Ma nelle immagini si hanno valori discreti quindi:

$$s_k = T(r_k) = \sum_{j=0}^k p_r(r_k) = \sum_{j=0}^k \frac{n_j}{n}$$

Quindi la forma di questa funzione dipende esclusivamente da $p_r(r)$ dell'immagine in input.

Fino ad adesso abbiamo considerato il caso normalizzato, ma nel caso reale un'immagine ha da 0 a $L-1$ valori possibili da poter registrare e ci sono $M \times N$ pixel di conseguenza la funzione sopra si riscrive come:

$$s_k = T(r_k) = (L-1) \cdot \sum_{j=0}^k p_r(r_k) = (L-1) \cdot \sum_{j=0}^k \frac{n_j}{MN} = \frac{(L-1)}{MN} \cdot \sum_{j=0}^k n_j$$

Questo algoritmo descritto è globale, nel senso che dopo che la funzione di mapping è compiuta allora è applicata a tutta l'immagine usando una **LookUp Table** a tutti i pixel dell'immagine.

Per ogni pixel nell'immagine originale, si usa $T(r)$ per trasformare il suo valore di intensità r nel nuovo valore s .

Delle volte, l'obiettivo è migliorare i dettagli di una foto grigia in delle piccole aree dell'immagine e quindi si usa una versione locale dell'algoritmo.

Quello che si fa è ottenere un **istogramma equalizzato locale** per ogni sliding window(*tile*) che si sposta in tutta l'immagine.

Per ogni tile dell'immagine si calcola l'istogramma equalizzato per fare il remapping dei pixel sul tile.

Ovviamente, questo processo è più costoso e si ha un output più rumoroso.

Però, questo approccio non è il migliore ma se ne ha un altro dove si vuole specificare la forma di un istogramma.

Questo metodo è chiamato **Histogram Matching** o **Histogram Specification**.

Histogram Matching

Task: Rimappare l'immagine I_d (discolored=sbiadito) tale che abbia lo stesso istogramma dell'immagine di riferimento J_r (reference).

Lo vogliamo fare, così si ha un metodo per restaurare un'immagine a partire da un'altra.

Poiché le immagini sono digitali non è generalmente possibile avere l'istogramma degradato coincidente a quello di riferimento.

Non posso copiare ed incollare l'istogramma della foto di riferimento perché:

1. **Distorta:** La relazione tra pixel vicini (importante per bordi e texture) verrebbe persa.
2. **Incoerente visivamente:** Potrebbero apparire artefatti o zone innaturali.
3. **Con possibile perdita di dettaglio:** Dettagli importanti dell'immagine originale potrebbero essere distrutti.

Come funziona Histogram Matching (passi chiave)

1. Calcolo delle CDF:

- Si calcola la funzione di distribuzione cumulativa (CDF) dell'immagine originale.
- Si calcola la CDF dell'immagine di riferimento (quella che ha l'istogramma desiderato).

2. Mappatura tra le due CDF:

- Si crea una funzione di mapping $T(r)$ che trasforma i valori di intensità dell'immagine originale r nei valori corrispondenti s dell'immagine di riferimento.
- Questo mapping si basa sul confronto delle due CDF.
- Dato che non siamo in dominio reale ma discreto non per forza si trova $CDF_2(s) = CDF_1(r)$ quindi bisogna trovare l' s tale che $CDF_2(s)$ è vicino al 1 (quello dell'immagine degradata).
- Quindi per creare l'immagine finale I , dall'immagine I_1 tale che abbia CDF come I_2 si fa così che:

$$I(x, y) = s \quad \text{sse} \quad CDF_1(r) > CDF_2(s - 1) \quad \text{e} \quad CDF_1(r) < CDF_2(s)$$

3. Trasformazione dei pixel:

- Ogni pixel dell'immagine originale viene trasformato secondo la funzione $T(r)$.

Spatial Operations

Processo Base:

Si definisce intorno ad un pixel (x,y) un quadrato. Quindi i vicini sono i pixel intorno a quel quadrato (*mask*). Il processo parte dall'angolo in alto a sinistra della foto e viene scannerizzato in orizzontale una riga alla volta.

L'operatore T viene applicato ad ogni locazione (x,y) con output $g(x,y)$. Questo operatore, però, utilizza per l'elaborazione i vicini della locazione passata.

Ma all'origine, parte dei vicini sono fuori dall'immagine.

Un operazione può essere quella della media dell'intensità dei suoi vicini, somma di tutti e 9 i valori diviso per 9.

Quindi:

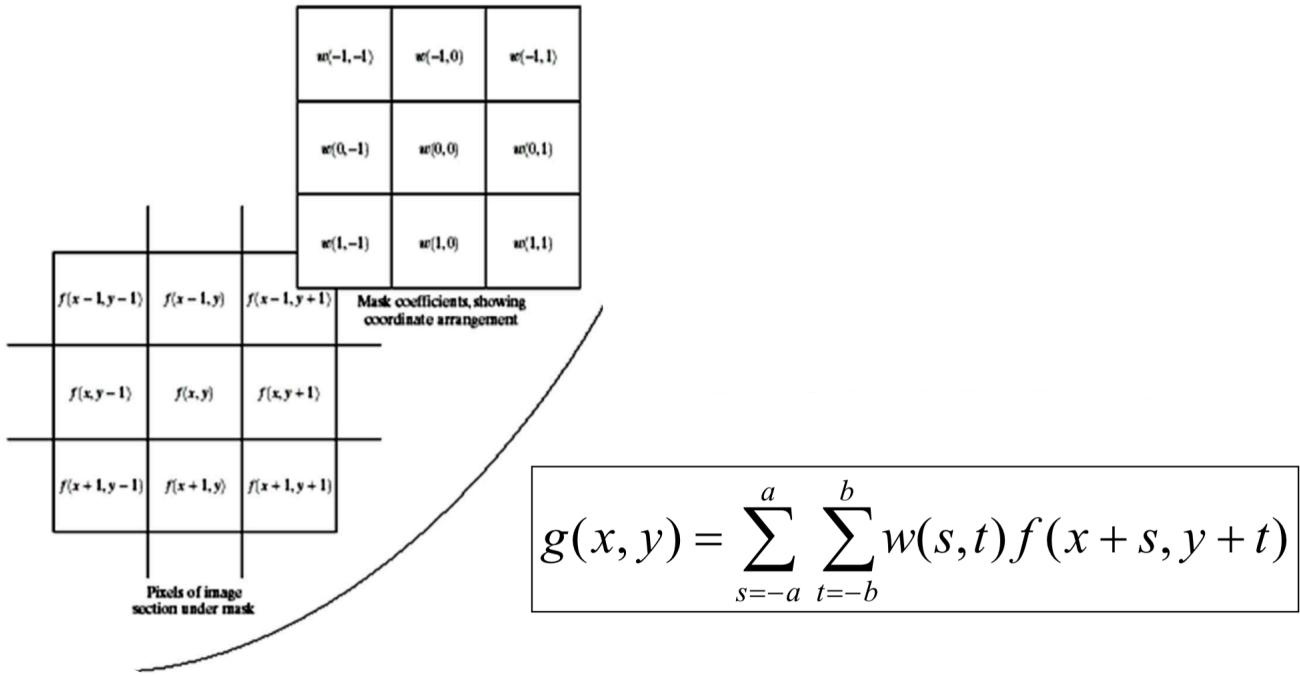
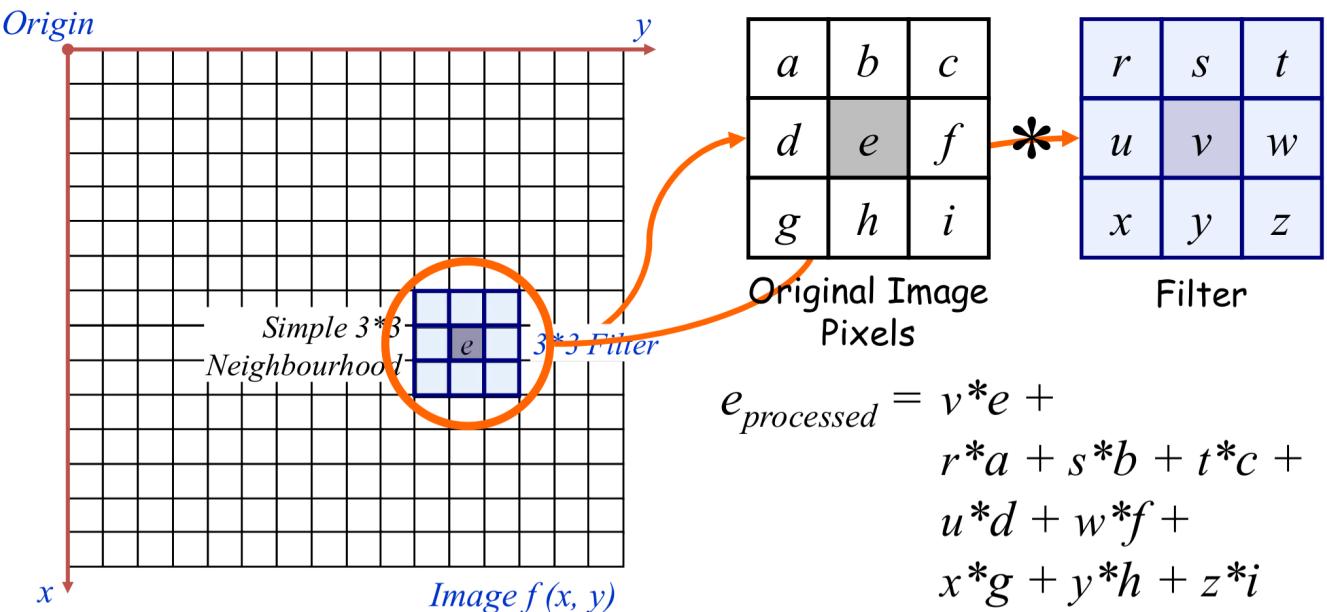
1. si definisce un punto di riferimento dell'immagine in input, $f(x_0, y_0)$
2. fa operazione che convolge tutti i suoi pixel vicini.
3. applica il risultato dell'operazione al pixel dell'immagine output con le stesse coordinate di riferimento per ogni pixel dell'immagine.
4. Ripeto per ogni pixel dell'immagine di input.

Si hanno due tipi di categorie:

- Lineari: uscita calcolata come somma di prodotti dei valori di pixel e coefficienti di maschera del vicinato del pixel di riferimento.
- Non Lineari: il pixel in output è selezionato da una sequenza ordinata di valori nel "vicinato" del pixel dell'immagine originale.

Filtri Lineari

Viene fatto il prodotto scalare tra il la mask dell'immagine originale e la mask del filtro.



Smoothing Linear Filters:

Si tratta di "lisciare" l'immagine in input facendo semplicemente la media intorno al valore centrale. Riduce il rumore ma fa un effetto di blur sui margini. La maschera è fatta tutta da coefficienti pari a $1/n$ dove n è la dimensione della maschera.

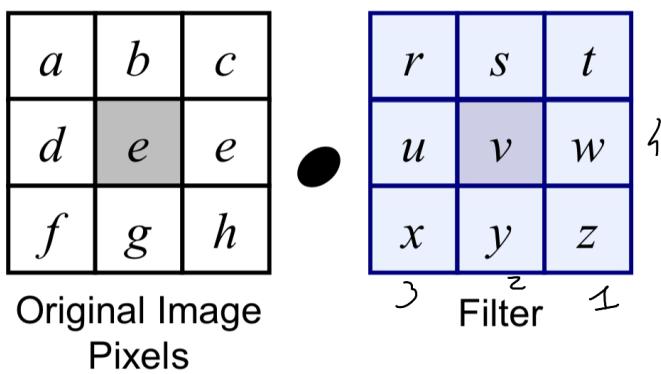
Più aumento la maschera e più si ha un effetto di blur.

Ma i pixel mancanti sui bordi come si possono risolvere?

- omettendo i pixel mancanti

- padding delle immagini con pixel bianchi o neri
- replicare i pixel dei bordi
- Troncare immagine

Fino ad adesso abbiamo visto come l'applicazione tra una maschera ed un gruppo attraverso la correlazione mentre adesso si parla di **convoluzione** come un'operazione simile intesa in questo modo:



$$e_{processed} = v^*e + \\ z*a + y*b + x*c + \\ w*d + u*e + \\ t*f + s*g + r*h$$

Per performare la convoluzione bisogna ruotare il filtro di 180 gradi e performare le stesse operazioni della correlazione. (I filtri simmetrici danno stesso risultato).

Ci sono anche dei Filtri che hanno alcuni elementi della maschera più importanti di altri. Ad esempio il pixel centrale ha più rilevanza e quindi ha più peso.

$$g(x, y) = \frac{\sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)}{\sum_{s=-a}^a \sum_{t=-b}^b w(s, t)}$$

Sharpening Spatial Filter or High pass Filter:

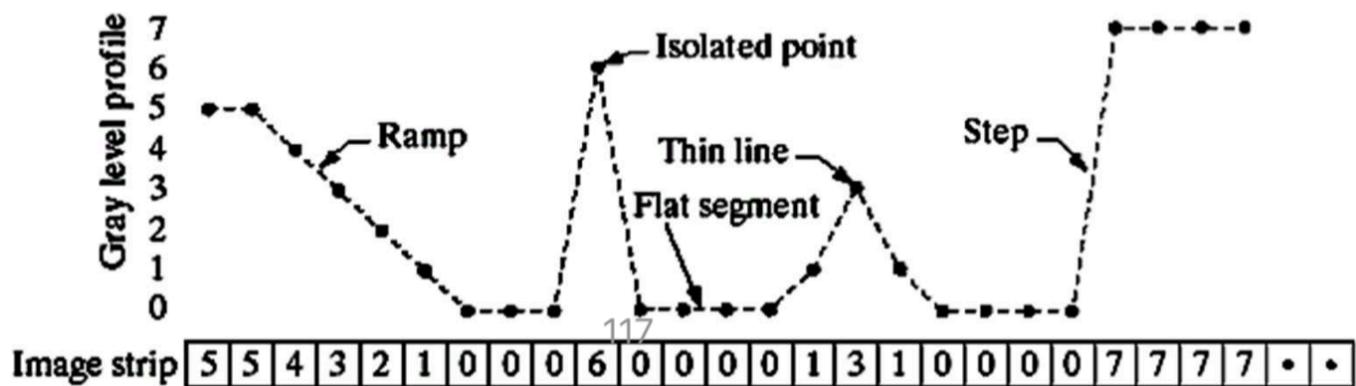
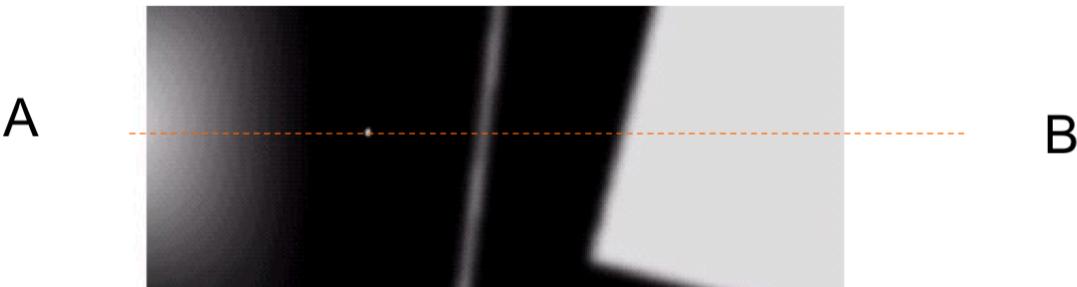
Questi tipi di filtri sono spaziali e il loro effetto è quello di preservare e enfatizzare le componenti di alto livello (come dettagli fini, punti, linee, margini), cioè per evidenziare le transizioni di intensità all'interno dell'immagine.

- Rimuovere la sfocatura da immagini
- Evidenziare i bordi

È basato sulla differenziazione dello spazio, la risposta di un operatore derivativo è proporzionale al grado di discontinuità dell'immagine.

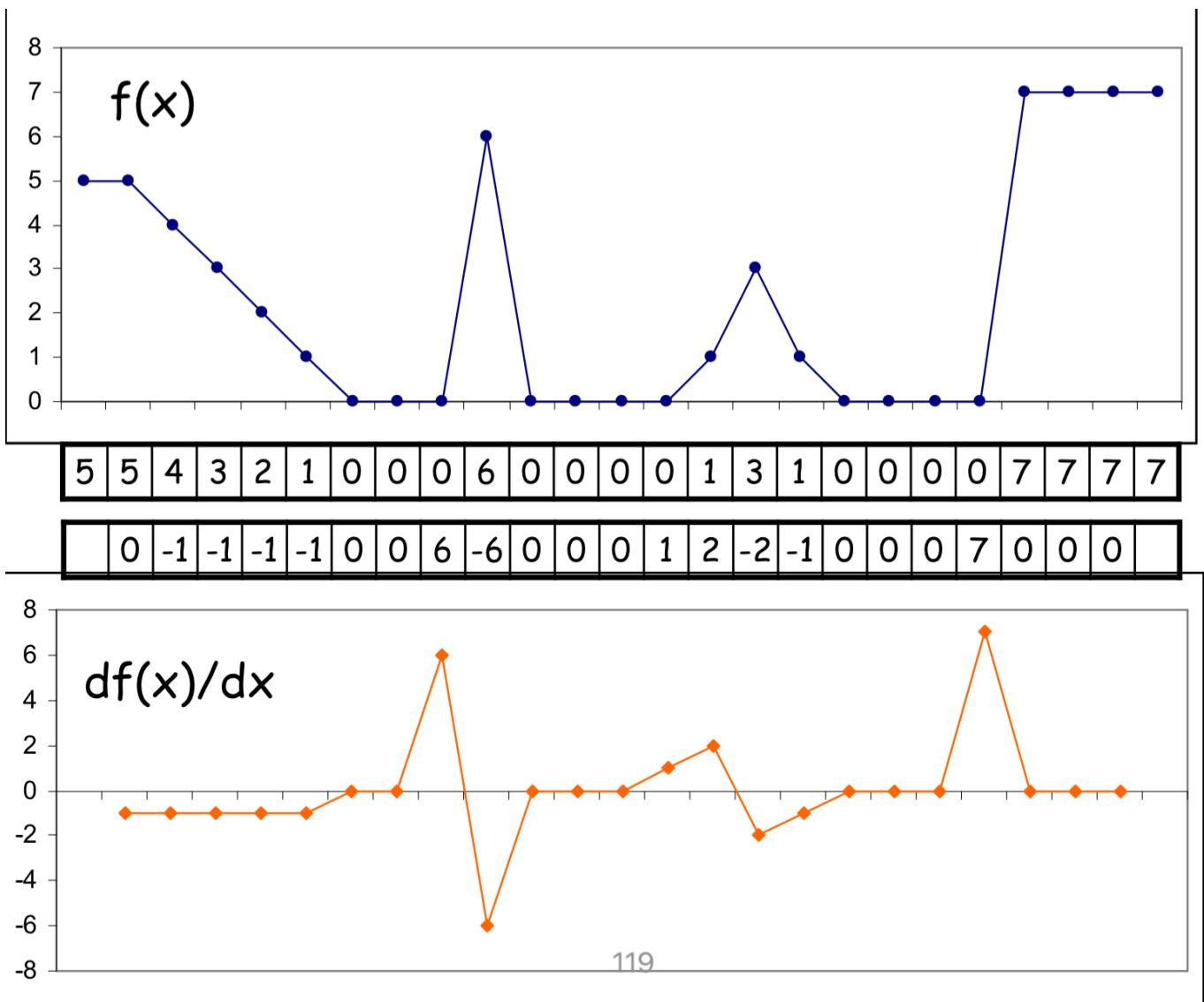
Le derivate di una funzione digitale sono definiti in termini di differenze (poichè valori discreti).

Esempio:



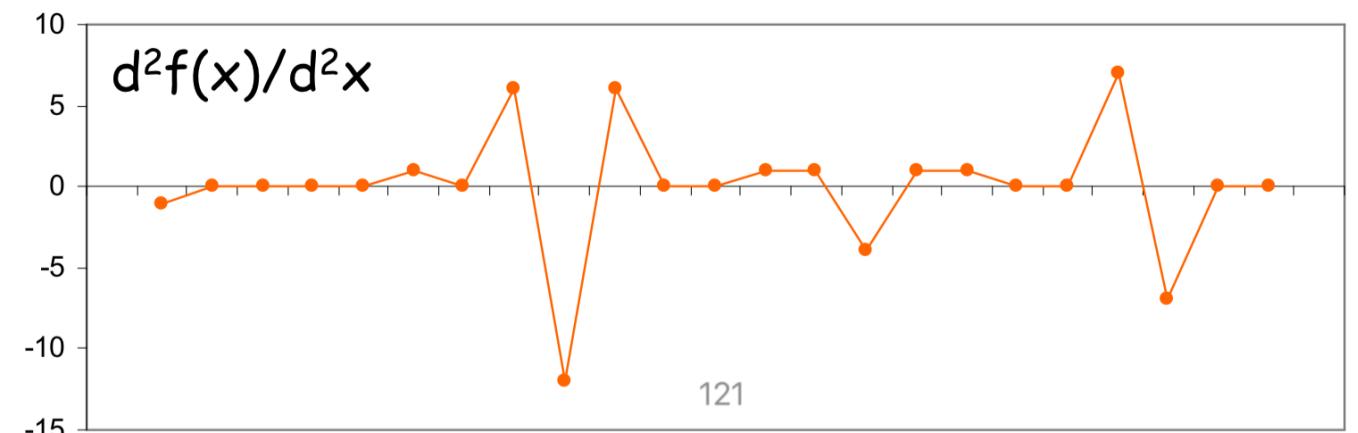
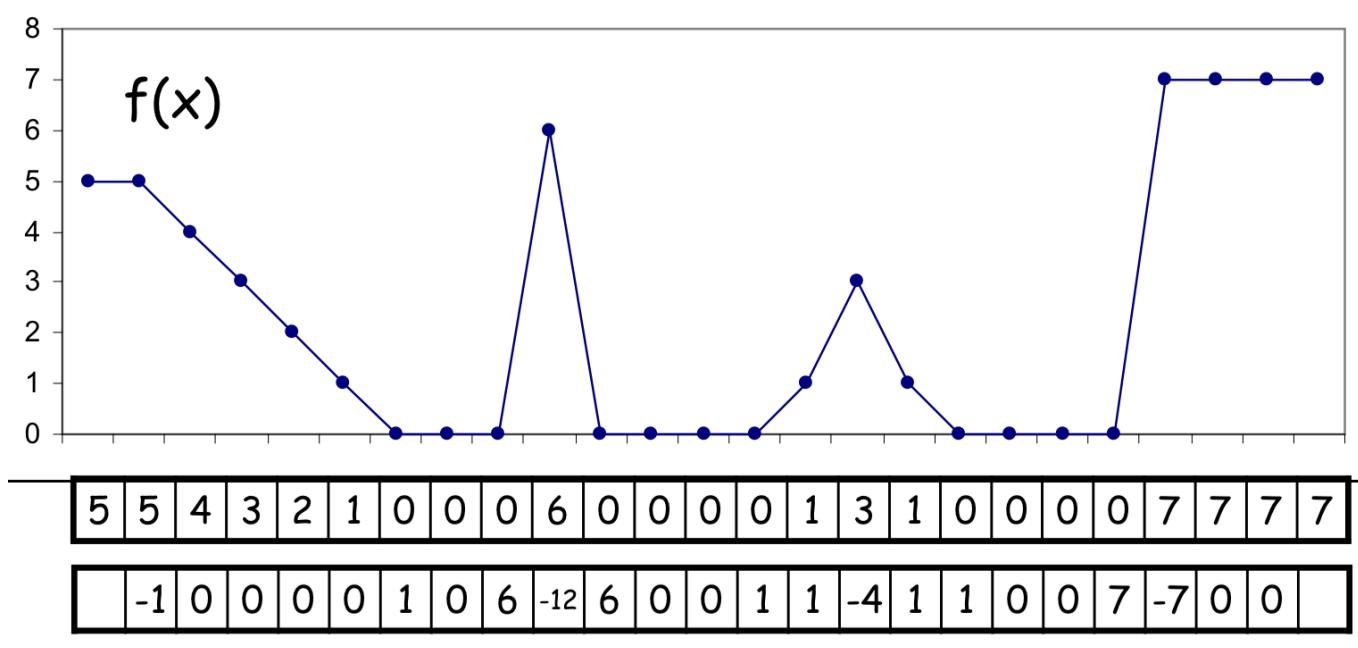
La formula per la prima funzione derivata è:

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



La seconda derivata ha questa formula

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



Si utilizza la derivata seconda perchè è più utile per la miglioria dell'immagine rispetto alla derivata 1:

- Ha una risposta più forte ai dettagli fini
- Implementazione più semplice

Un filtro sharpening è la Laplaceana definita come:

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

dove ricordiamo che:

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

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

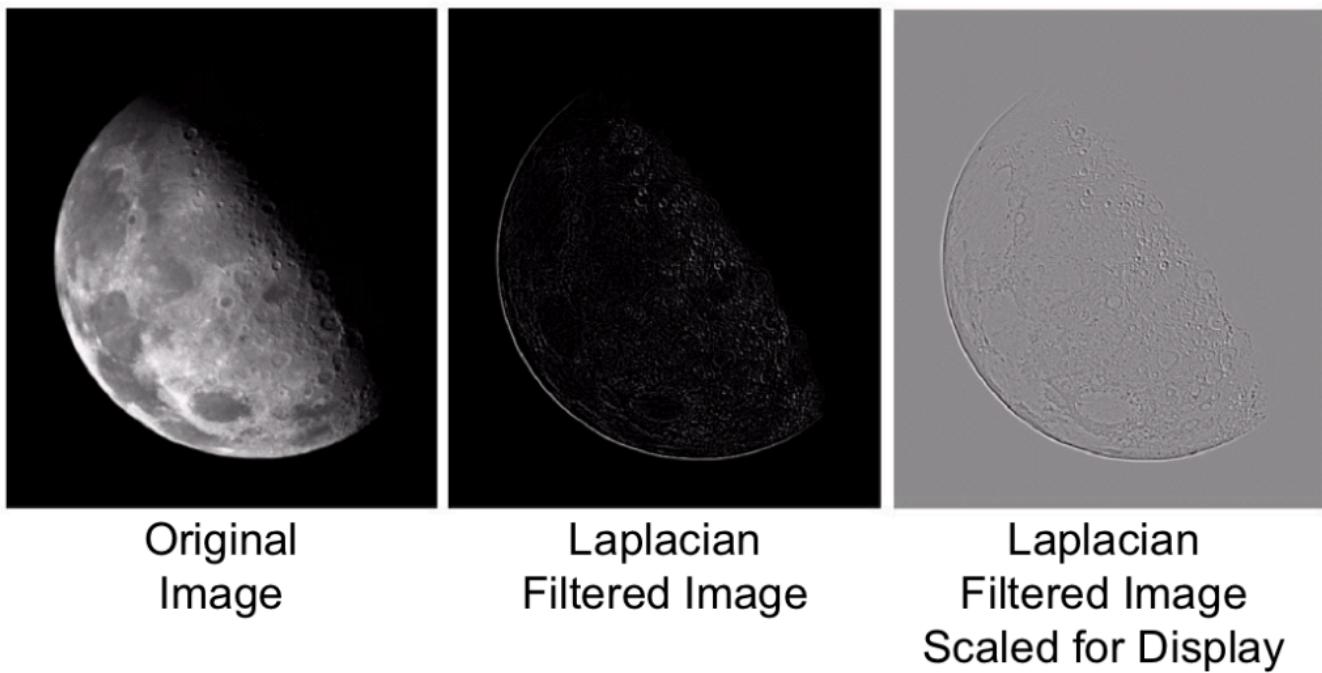
Di conseguenza:

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

vediamo che da qui possiamo ricondurci ad una correlazione con il Filtro Laplaciano creato così:

0	1	0
1	-4	1
0	1	0

Applicato questo filtro ad un'immagine, abbiamo una nuova immagine con evidenza nei margini e nelle discontinuità.

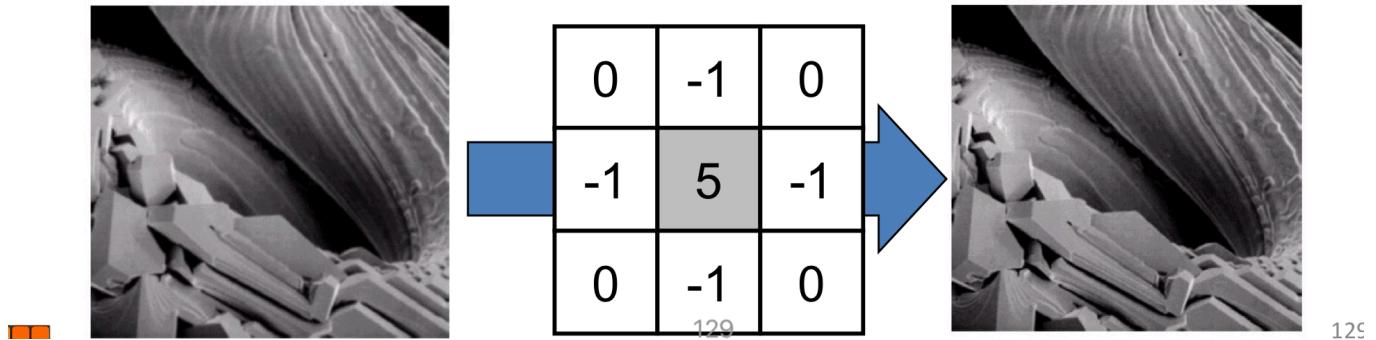


Vediamo da queste foto che non si ha un'immagine migliorata, perché la laplaciana è un operatore derivativo, perché dove il contenuto è costante si ha zero.

Quindi quello che dobbiamo fare è sottrarre l'immagine:

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

$$\begin{aligned} g(x, y) &= f(x, y) - \nabla^2 f \\ &= f(x, y) - [f(x+1, y) + f(x-1, y) \\ &\quad + f(x, y+1) + f(x, y-1) - 4f(x, y)] \\ &= 5f(x, y) - f(x+1, y) - f(x-1, y) - f(x, y+1) - f(x, y-1) \end{aligned}$$



12c

Filtro di Sobel

Ci sono altri tipi di filtri Laplaciani, vediamone altri quindi ad esempio utilizzando l'ampiezza del gradiente:

$$M(x, y) = \text{mag}(\nabla f) = \left[g_x^2 + g_y^2 \right]^{\frac{1}{2}} = \left[\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2 \right]^{\frac{1}{2}}$$

Isotropic !

For practical reasons this can be simplified as:

$$M(x, y) \approx |g_x| + |g_y|$$

Not isotropic !

Usually computed as:

$$\begin{aligned} M(x, y) &\approx \left| (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \right| \\ &\quad + \left| (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \right| \end{aligned}$$

134

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

134

Si dice che il filtro Laplaciano è isotropico se calcola le variazioni in tutte le direzioni.

Le seguenti equazioni sono derivate con gli **operatori di Sobel**.

-1	-2	-1
0	0	0
1	2	1

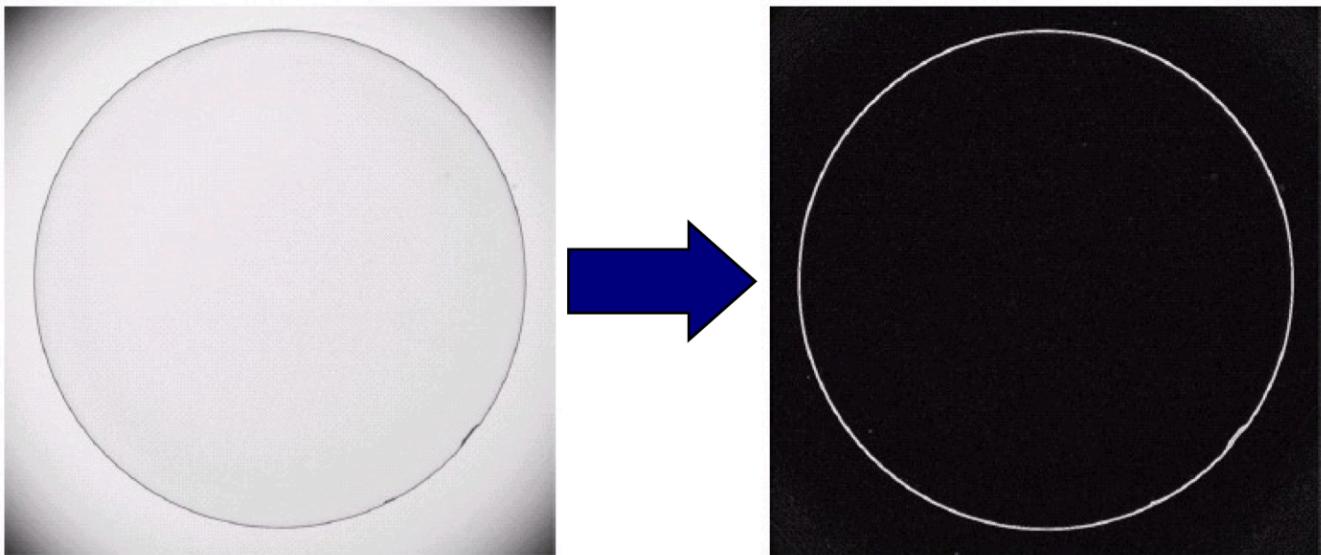
-1	0	1
-2	0	2
-1	0	1

$$g_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$g_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

Per filtrare l'immagine si utilizzano tutti e due i filtri e i risultati vengono sommati (in valore assoluto).

Questi filtri sono chiamati Filtri di Sobel e servono per evidenziare i margini.



Filtri Non Lineari

I filtri non lineari sono basati su operazioni non lineari che comprendono i pixel vicini. Il concetto di maschera non è chiaro come nei filtri lineari, qui si deve vedere come una funzione non lineare che opera con i pixel vicini a quello di riferimento e la quale risposta influenza il centro del vicinato di pixel.

Infatti la classe di questi filtri non lineari è basata su l'ordinamento di pixel contenuti nell'area racchiusa del pixel.

Il valore del pixel centrale è rimpiazzato con il valore dell'ordinamento.

Degli esempi sono: filtro mediano, max e min.

Filtro Mediano:

Sostanzialmente quello che si fa è rimpiazzare il valore del pixel con il mediano dei valori nel

vicinato. Con mediano non si intende la media bensì si riordinano tutti i valori del vicinato e si prende quello che sta nel mezzo.

Ad esempio: se il vicinato è una matrice 3×3 allora il mediano sarà il quinto elemento.

La funzione mediana forza i punti con livelli distinti di grigio a essere come i vicini.

Questo permette una grande gestione del rumore con meno blur rispetto ai filtri di smoothing lineari.

Dominio di Frequenza:

Iniziamo con una definizione importante, nel quale si dice:

"Qualsiasi segnale periodico, può essere espresso come somma di seni e/o coseni con differenti frequenze, moltiplicate per differenti coefficienti (Serie di Fourier). Anche i segnali non periodici possono essere rappresentati come l'integrale di seni e/o coseni a differenti frequenze, moltiplicate per differenti coefficienti di Fourier.(Trasformata di Fourier)."

Qualsiasi segnale, rispetto al tempo/ spazio / o qualsiasi altra variabile, può essere rappresentata come combinazione di funzioni armoniche con diverso periodo.

Il Processo di Fourier si focalizza nel trovare la relazione tra contenuti armonici del segnale di output e quello di input.

Nella rappresentazione di Fourier, termini armonici con frequenza alta servono a ricostruire cambiamenti rapidi del segnale. Viceversa se si hanno termini armonici con frequenza bassa servono a costruire cambiamenti tenui.

Quindi il dominio di frequenza e quello di spazio sono tra loro reciproci poichè se si hanno cambiamenti piccoli nel dominio dello spazio allora si hanno valori larghi nel dominio di frequenza.

Idea Principale: La Trasformata di Fourier in due dimensioni serve a far sì che la **combinazione di funzioni basi armoniche 2-D scalate e shiftate** possano sintetizzare una **funzione spaziale**.

Ripasso di Segnali:

La trasformata di Fourier si ricorda essere questa:

$$F(f) = F(f(t)) = \int_{-\infty}^{\infty} f(t) \cdot e^{-j2\pi ft} dt$$

Dove si nota che $F(f)$ è un numero complesso quindi è rappresentato attraverso:

- **ampiezza**: peso per ogni componente di frequenza.
- **fase**: quando/dove le componenti della frequenza occorre.

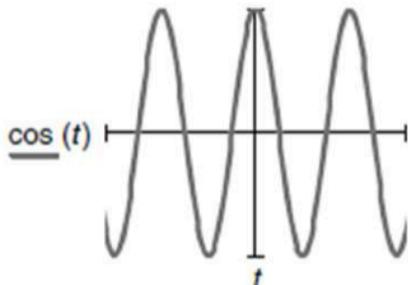
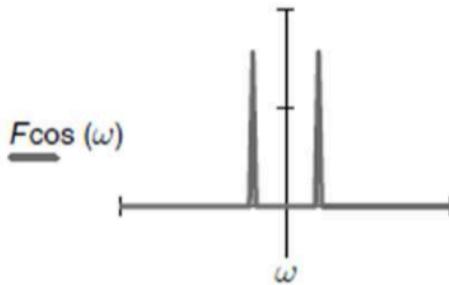
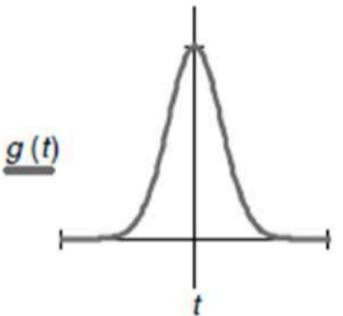
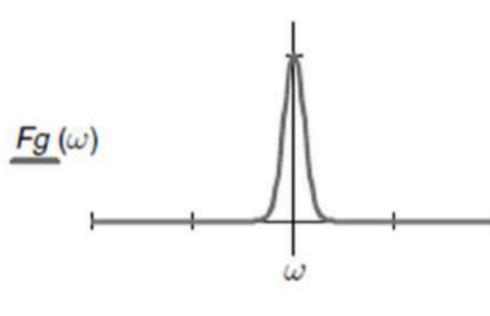
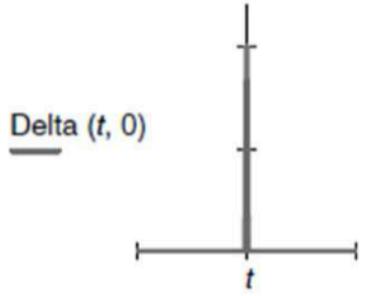
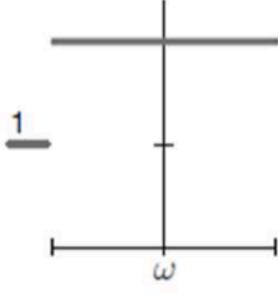
$$F(f) = R(f) + j \cdot I(f)$$

$$F(f) = |F(f)| \cdot e^{j\theta(f)}$$

Si ha anche l'inversa:

$$f(t) = \int_{-\infty}^{\infty} F(f) \cdot e^{+j2\pi ft} df$$

Coppie di Trasformate:

Time domain signals	Frequency domain spectra
 (a) Cosine wave	 (b) Fourier transform of cosine wave
 (c) Gaussian function	 (d) Spectrum of Gaussian function
 (e) Delta function	 (f) Frequency content of delta function

Il fatto, però, è che le nostre immagini non hanno valori continui ma sono rappresentati come funzioni discrete.

Possiamo supporre che sono stati derivati da un segnale continuo attraverso l'operazione di campionamento.

Quindi, prendiamo in ingresso un segnale continuo che verrà campionato ogni ΔT , di conseguenza i valori di un segnale continuo vengono presi ogni ΔT rendendo il segnale discreto. Questo è la base per un convertitore da analogico a digitale.

Matematicamente si può spiegare così:

$$\tilde{f}(t) = f(t) \cdot s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} f(t) \cdot \delta(t - n\Delta T)$$

Il fatto è che questo campionamento deve essere fatto in modo giusto affinchè il comportamento della funzione sia descritto. Infatti utilizzando ΔT grandi si potrebbe trascurare comportamenti diversi della funzione...

Possiamo capire meglio il processo nel dominio della frequenza.

Teorema della Convoluzione: La trasformata del prodotto di due segnali corrisponde alla convoluzione dello spettro dei due segnali.

Convoluzione: operazione che consiste nel prende lo spettro di uno, fliparlo rispetto ad asse y e muoverlo rispetto al orientamento contrario dell'asse x.

Lo spettro del campionamento è:

$$S(f) = \frac{1}{\Delta T} \cdot \sum_{n=-\infty}^{\infty} \delta(f - \frac{n}{\Delta T})$$

Quindi:

$$\begin{aligned} \tilde{F}(f) &= F(f) \otimes S(f) = \int_{-\infty}^{\infty} F(\mu) \cdot S(f - \mu) d\mu \\ &= \int_{-\infty}^{\infty} F(\mu) \cdot \frac{1}{\Delta T} \cdot \sum_{n=-\infty}^{\infty} \delta(f - \mu - \frac{n}{\Delta T}) d\mu = \frac{1}{\Delta T} \sum_{n=-\infty}^{\infty} \int_{-\infty}^{\infty} F(\mu) \cdot \delta(f - \mu - \frac{n}{\Delta T}) d\mu \end{aligned}$$

Quest'ultima parte attraverso la proprietà di campionamento della Delta di Dirac si riconduce a:

$$\tilde{F}(f) = \frac{1}{\Delta T} \sum_{n=-\infty}^{\infty} F(f - \frac{n}{\Delta T}).$$

Di conseguenza, il risultato del campionamento nel dominio di frequenza è un altro segnale periodico infinito e continuo.

Scriviamo in questo altro modo il risultato ottenuto:

$$\tilde{F}(f) = f_s \sum_{n=-\infty}^{\infty} F(f - n \cdot f_s)$$

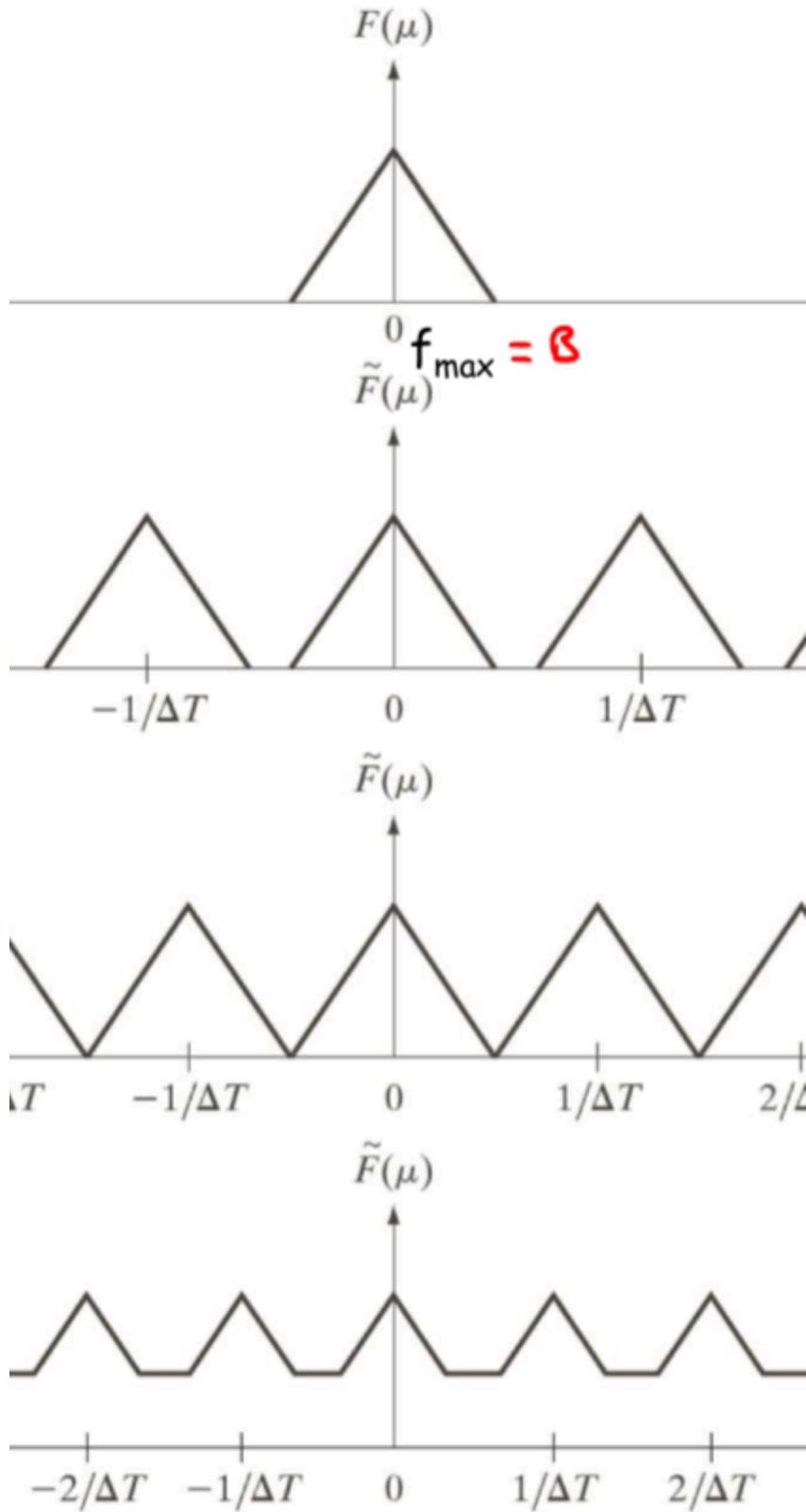
Quindi la trasformata di Fourier viene ripetuta ogni f_s , tra una trasformata e l'altra si ha una spaziatura di f_s .

Ogni Trasformata avrà una lunghezza di banda $B(-f_{max}, f_{max})$. Se quindi la frequenza di campionamento f_s è maggiore del doppio di f_{max} allora si ha il fenomeno detto

oversampling(non si ha collisione).

Invece se è minore di $2 \cdot f_{max}$ si ha la collisione tra i segnali della trasformata e quindi si ha il fenomeno dell' *undersampling*.

Se è uguale si ha il *critically-sampling*.



Teorema del Campionamento:

Un segnale continuo a banda limitata può essere totalmente ricostruito da i suoi campionamenti se essi sono stati acquisiti ad una frequenza pari o maggiore di due volte la più alta frequenza della trasformata.

$$f_s > 2f_{max}$$

$$1/\Delta T > 2f_{max} \rightarrow \Delta T < 1/(2f_{max})$$

Questo criterio è chiamato **Criterio di Nyquist**.

Si può rispettandolo ricostruire il segnale originale filtrando lo spettro con un filtro passa basso:

$$F(f) = H(f) \cdot \tilde{F}(f)$$

Una volta filtrato possiamo ricostruire il segnale attraverso l'inversa della trasformata di fourier.

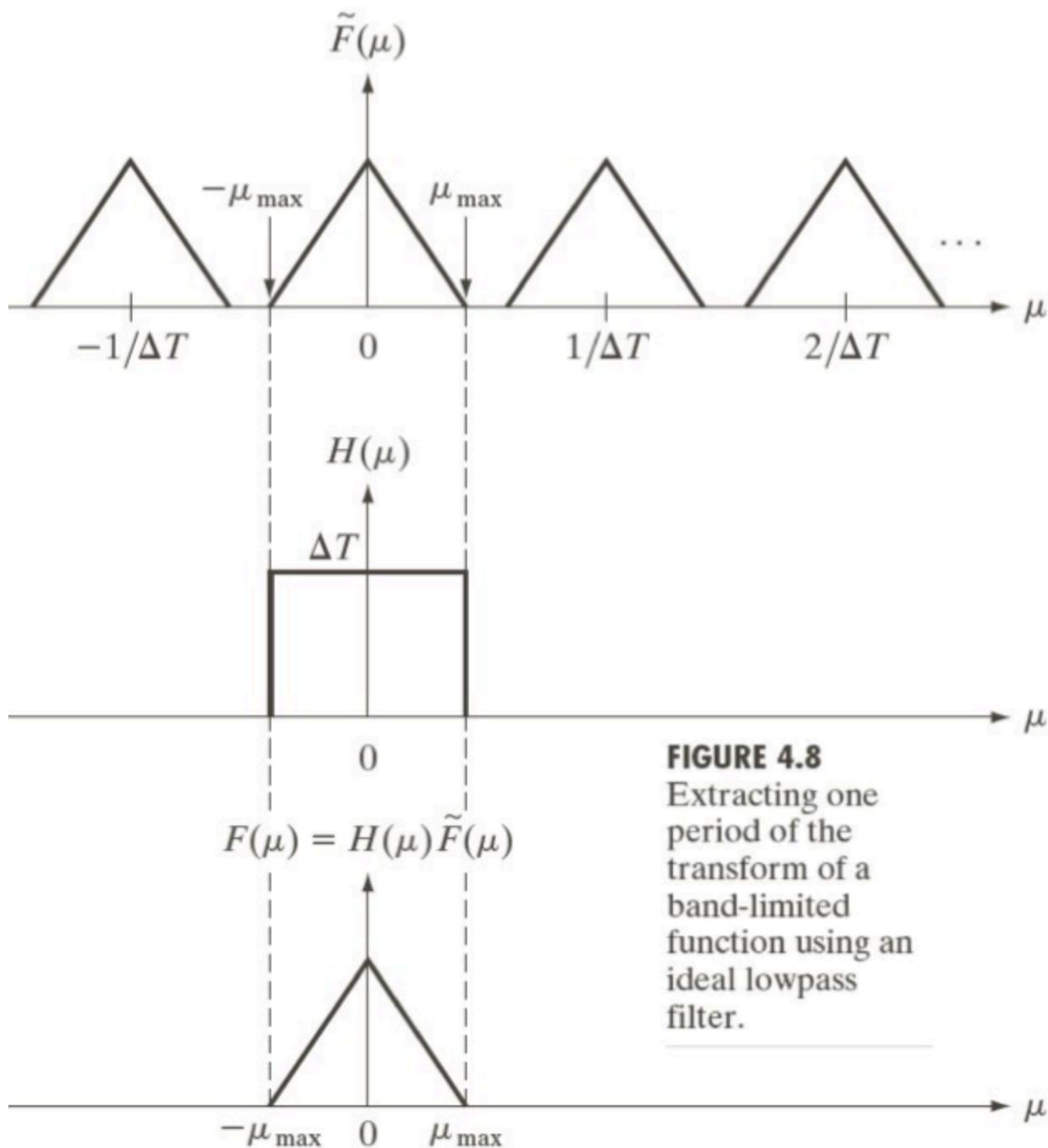


FIGURE 4.8
Extracting one period of the transform of a band-limited function using an ideal lowpass filter.

Se si dovesse fare un *undersampling* del segnale, nel momento del filtraggio si otterrebbe uno spettro diverso dalla natura dello spettro campionato, avendo e ricostruendo quindi un segnale sbagliato.

Questo effetto è chiamato **Aliasing**.

2D-DFT:

Estendiamo, quindi, la trasformata di Fourier su due dimensioni e questa trasformata $F(u,v)$ viene definita così:

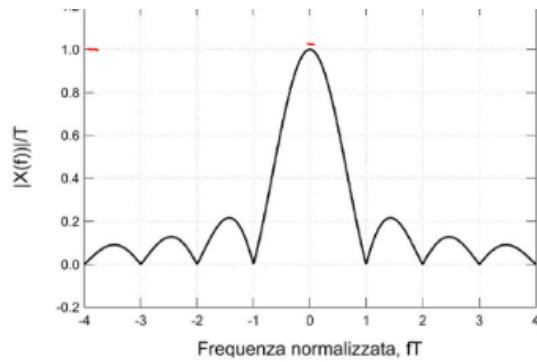
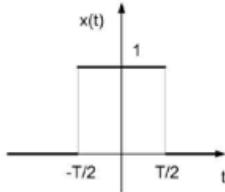
$$F(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) e^{-j2\pi(xu+yv)} dx dy$$

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u, v) e^{+j2\pi(xu+yv)} du dv$$

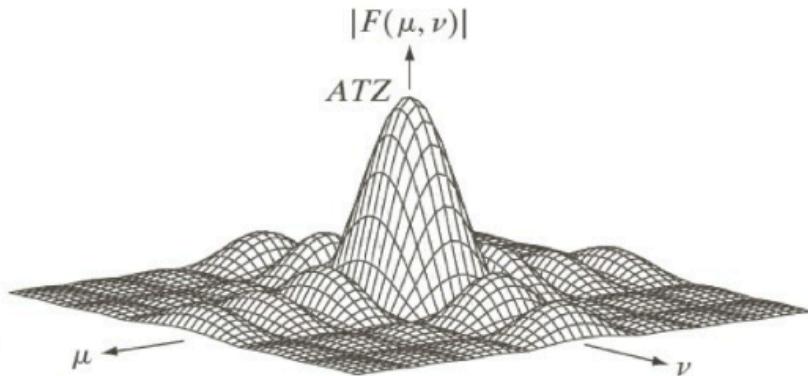
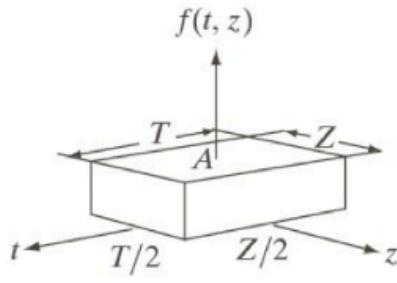
dove:

- $F(u, v)$ è un numero complesso
- L'immagine è decomposta in una somma di sinusoidi spaziali con differenza ampiezza, fase e direzione.

$$\text{1D} \quad \text{rect}\left(\frac{t}{T}\right) \Leftrightarrow T \cdot \text{sinc}(fT)$$



2D :



In maniera simile a 1-D, bisogna campionare in due dimensioni che può essere moltiplicare $f(x,y)$ per una funzione di campionamento (2-D impulse train):

$$s_{\Delta X, \Delta Y} = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \delta(x - n\Delta X, y - n\Delta Y)$$

Da qui si può ricavare il teorema del campionamento 2-D che dice: un segnale 2D continuo a banda limitata $f(x,y)$ può essere ricostruito completamente senza errori se un insieme dei suoi campionamenti vengono campionati con un intervallo pari a:

$$\Delta X < 1/2u \quad e \quad \Delta Y < 1/2v$$

Altrimenti si ha l'Aliasing.

Questo tipo di trasformata è una trasformata di $M \times N$ pixel con la locazione spaziale indicizzata attraverso x e y mentre le due frequenze u e v , sono le frequenze orizzontali e verticali.

$$F(u, v) = F_{u,v} = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f_{x,y} e^{-j2\pi(ux/M + vy/N)}$$

$$f(x, y) = f_{x,y} = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F_{u,v} e^{+j2\pi(ux/M + vy/N)}$$

dove i parametri M e N sono rispettivamente il numero di campioni sull'asse x e sull'asse y della immagine.

M e N derivano dal fatto che l'immagine (o il segnale) è discretizzata e ha dimensioni finite. Questi parametri non compaiono nelle equazioni delle trasformate continue perché il dominio continuo non necessita di limiti esplicativi.

Basi DFT:

Data un'immagine $f(x,y)$ con $x = 0, \dots, N - 1$ e $y = 0, \dots, M - 1$ allora la funzione base DFT:

$$B_{u,v}(x, y) = \frac{1}{\sqrt{MN}} e^{j2\pi(\frac{ux}{N} + \frac{vy}{M})}$$

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \cdot B_{u,v}(x, y)$$

$F(u, v)$ è un numero complesso che può essere decomposto in parte reale e immaginaria o in forma polare con ampiezza e fase.

$$F(u, v) = |F(u, v)| \cdot e^{j\phi(u, v)}$$

Dove l'ampiezza è la radice della somma dei quadrati di $R(u, v)$ e $I(u, v)$.

Mentre la fase è:

$$\phi(u, v) = \arctan\left[\frac{I(u, v)}{R(u, v)}\right]$$

Se $f(x, y)$ è un segnale reale (come un'immagine) $\rightarrow F(u, v)$ è simmetrico coniugato.

Quindi lo spettro della magnitudine è simmetrico rispetto all'origine.

$$|F(-u, -v)| = |F(u, v)|$$

Mentre la fase è anti-simmetrico rispetto all'origine:

$$\phi(-u, -v) = -\phi(u, v)$$

Nell'origine 2D-DFT nell'origine è uguale alla media dell'immagine:

$$F(0, 0) = \frac{1}{MN} \cdot MN \sum_x \sum_y f(x, y) \cdot e^{-j2\pi(\frac{ux}{M} + \frac{vy}{N})} = MN \cdot \frac{1}{MN} \sum_x \sum_y f(x, y) \cdot e^0$$

$$F(0, 0) = MN \cdot \bar{f}(x, y)$$

Poichè MN è un valore grosso , $|F(0,0)|$ è il componente più grosso e viene chiamato **Componente DC** della trasformata.

La trasformata di Fourier di 1-D e 2-D sono infinitamente periodiche quindi si può trovare:

$$F(u, v) = F(u + k_1 M, v) = F(u, v + k_2 N) = F(u + k_1 M, v + k_2 N)$$

È conveniente alcune volte avere nell' intervallo un periodo della trasformata completo propriamente e lo si può fare tramite la trasformazione di frequenza.

$$f(x)e^{j2\pi(u_0x/M)} \leftrightarrow F(u - u_0)$$

Se $u_0 = M/2$ allora viene che:

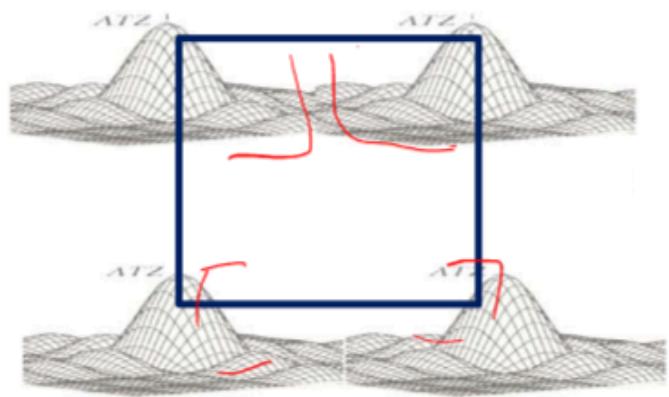
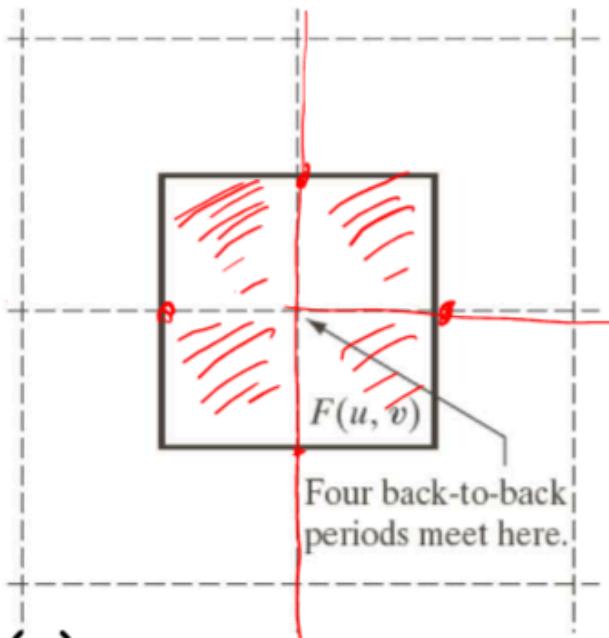
$$f(x)e^{j2\pi(M/2)x/M} = f(x) \cdot e^{j\pi x} = f(x) \cdot (e^{j\pi})^x = f(x) \cdot (-1)^x$$

Quindi:

$$f(x) \cdot (-1)^x \leftrightarrow F(u - M/2)$$

Quindi attraverso -1^x si riesce a shiftare i dati affinche l'ampiezza massima $|F(0)|$ sta a $u=M/2$.

Ricollegandoci, quindi, alle trasformate 2D- DFT esse hanno 4 quarti di periodo in ogni vertice della foto (data dalla periodicità della trasformata).

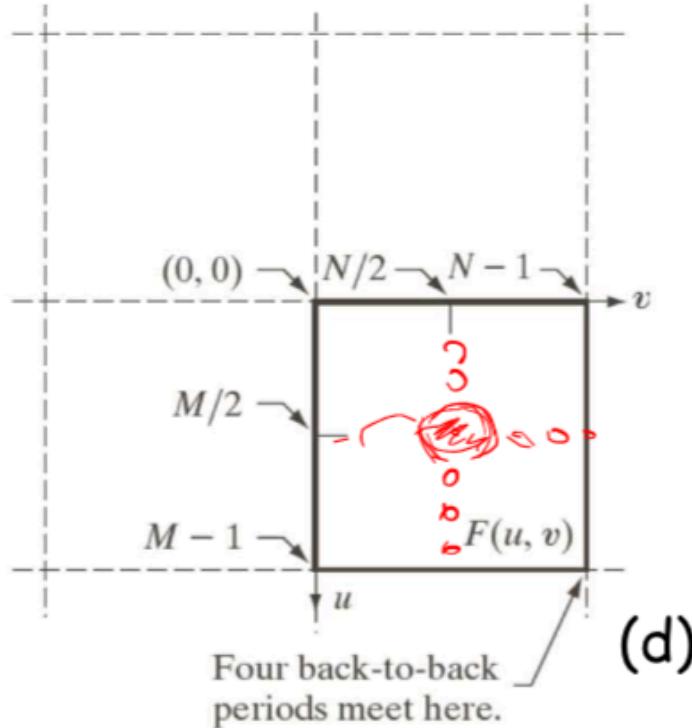
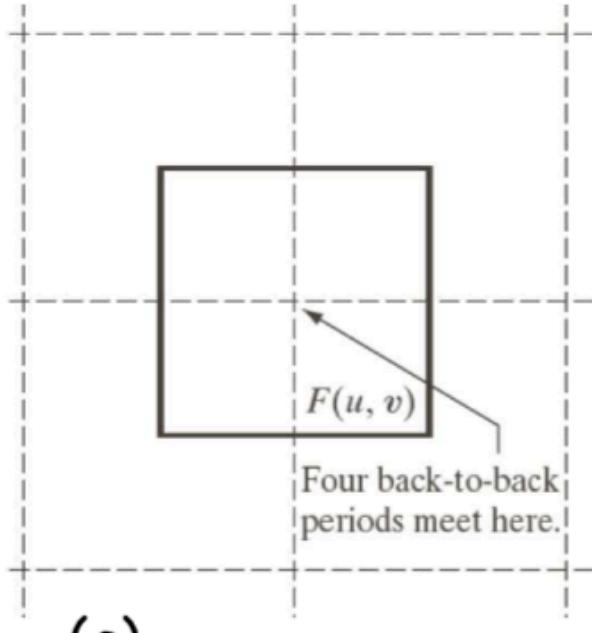


Quello che si può fare affinchè si riesca ad in quadrare solo un periodo rispetto a quattro quarti di periodo è shiftare di $M/2$ e $N/2$ e lo si fa attraverso:

$$f(x, y)e^{j2\pi(u_0x/M+v_0y/N)} \leftrightarrow F(u - u_0, v - v_0)$$

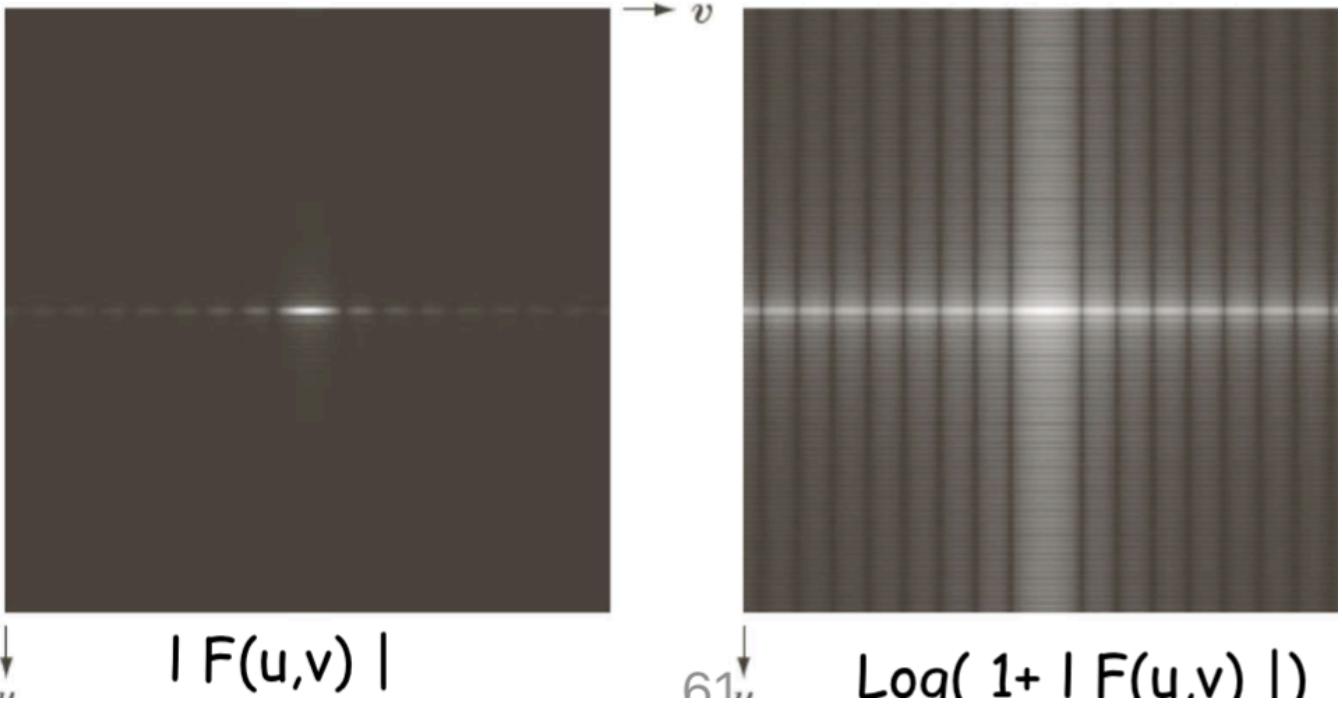
quindi:

$$f(x, y)(-1)^{x+y} \leftrightarrow F(u - M/2, v - N/2)$$



(c)

(d)



Vediamo da questa foto, che come nel dominio spaziale il Logaritmo aumenta il range dei valori chiari.

2D- DFT Traslazione e Rotazione:

Se si usano le coordinate polari:

$$x = r \cdot \cos\theta \quad y = r \cdot \sin\theta \quad u = \omega \cos\phi \quad v = \omega \sin\phi$$

Allora le trasformate, diventano questo:

$$f(r, \theta) \leftrightarrow F(\omega, \phi)$$

e avviene che:

$$f(r, \theta + \theta_0) \leftrightarrow F(\omega, \phi + \theta_0)$$

Questo vuol dire che se si ruota $f(x,y)$ di un certo angolo allora anche $F(u,v)$ ruota dello stesso angolo e viceversa.

Mentre lo spettro è insensitivo alle transazioni:

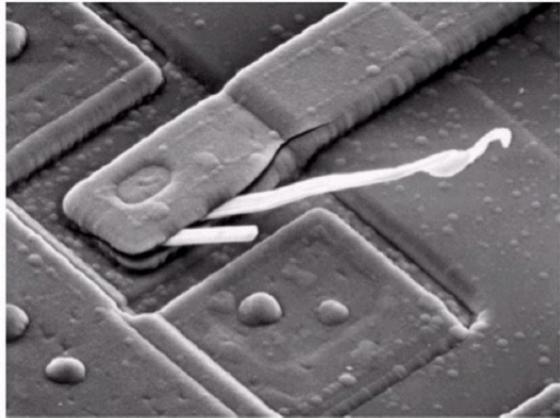
$$f(x - x_0, y - y_0) \leftrightarrow e^{-j2\pi(vx_0 + uy_0)} \cdot F(u, v)$$

Ma l'ampiezza di questa traslazione è sempre uguale a $|F(u,v)|$.

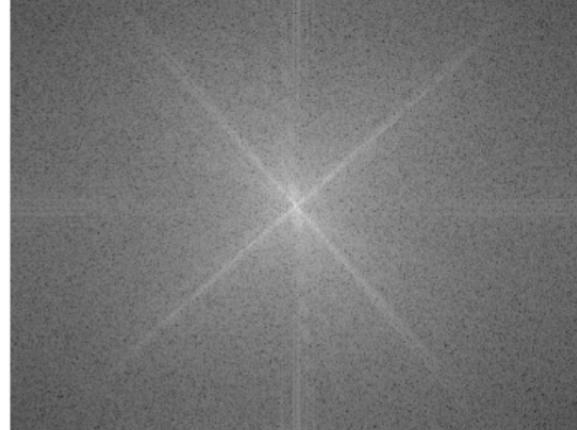
Importante: Le regioni luminose nell'immagine dell'ampiezza della DFT corrispondono alle frequenze che hanno un grosso magnitudo nell'immagine reale

Relazioni tra l'immagine e la trasformata di Fourier

- $F(u,v)$ contiene la somma di tutti i valori di $f(x,y)$ moltiplicati per un esponenziale.
- a parte alcuni casi, è impossibile fare delle associazioni dirette tra le specifiche componenti dell'immagine e i suoi coefficienti della trasformata.
- La frequenza($u=0, v=0$) corrisponde alla media del grigio nella foto.
- Se ci allontaniamo dall'origine, le frequenze basse rappresentano le componenti di un'immagine che variano lentamente.
- Mentre ad altre frequenze corrispondono a cambiamenti repentini di livelli di grigio (come margini di un oggetto).



77



L'immagine (spazio spaziale) e lo spettro di Fourier (spazio delle frequenze) sono strettamente correlati.

Le strutture visibili nell'immagine generano frequenze specifiche nello spettro:

- Dettagli netti e bordi → Frequenze alte.
- Orientamento dei bordi → Direzione delle frequenze nello spettro.
- La componente DC è probabilmente quella al centro e questo vuol dire che è stato fatto uno shift al centro.

Infatti vediamo come il margine dell'oggetto che è orientato di $\pm 45^\circ$ sono caratteristici di quelle componenti prominenti diagonali nello spettro di Fourier.

Questo perché abbiamo detto che i confini netti scaturiscono valori intensi nello Spettro.

È importante capire, che lo spettro della fase è **molto più importante** dell'ampiezza.

Infatti questa è quella che determina le caratteristiche della forma nell'immagine.

Ma se si prende uno spettro della fase non possiamo riconoscere niente che possa essere associato alla figura.

Se utilizzassi IDFT usando lo spettro di ampiezza di un rettangolo e lo spettro in fase di una donna, il risultato sarebbe molto più simile alla donna che al rettangolo.

Effetto strano

Ma perchè si ha alta energia nelle componenti orizzontali e verticali?

Questo è **dovuto** alla ripetizione delle immagini , poichè stiamo campionando sia nel dominio di frequenza che in quello spaziale e abbiamo repliche in entrambi i domini.



I bordi delle immagini, infatti, provocano una falsa discontinuità.

Quando catturi un'immagine (ad esempio, con una fotocamera digitale), stai eseguendo un **campionamento reale**:

- Una scena continua viene campionata in una griglia di pixel discreti.
- Questo è il campionamento **spaziale originale** e determina la dimensione $M \times N$ dell'immagine (numero di pixel in altezza e larghezza).

Tuttavia, questo campionamento reale **non crea repliche**; produce un'immagine discreta che rappresenta solo una porzione della scena reale.

Qui entra in gioco il **teorema di campionamento**:

- Quando prendi un'immagine discreta (cioè campionata nello spazio continuo), il suo **spettro continuo** diventa **periodico**.
- Inversamente, quando prendi una **immagine discreta** e applichi la DFT, si assume che sia **periodica nello spazio**.

Quindi, la periodicità spaziale che porta alle repliche **non è direttamente causata dal campionamento fisico dell'immagine originale**, ma è un effetto del modello della DFT, che

assume periodicità nei dati discreti.

Quando calcoli la DFT su un'immagine:

- **Non crei esplicitamente delle repliche nello spazio.**
- Tuttavia, la DFT assume che l'immagine sia periodica, e questo si manifesta nei risultati:
 - Nel dominio spaziale, l'immagine si ripete periodicamente.
 - Nel dominio delle frequenze, lo spettro è periodico e ripetuto ogni N .

Algoritmi di Interpolazione

Con questi algoritmi cerchiamo approssimazioni tramite interpolazioni tra campionamenti.

Gli algoritmi di Interpolazione possono essere raggruppati in due categorie:

- Adattivi: questi metodi cambiano dipendendo da cosa si sta interpolando.
- Non Adattivi: metodi trattano pixel in modo equo. Ci sono algoritmi proprietari e questi applicano diverse versioni dell'algoritmo quando trovano la presenza di un margine con l'obiettivo di minimizzare gli artefatti di interpolazione.

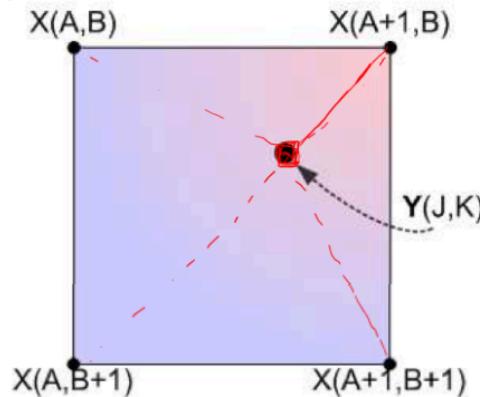
#RIGUARDARE !

Nearest-neighbor Interpolation:

Il metodo più basico e che richiede meno tempo poichè viene considerato un solo pixel, ovvero quello più vicini al punto da interpolare, e non vengono considerati gli altri vicini. Serve a far diventare il pixel solo più grande.

Si fa questo controllo attraverso questa concatenazione di IF...

```
IF ( K-B < B+1-K )
    Pixel is one of the top two
ELSE
    Pixel is one of the bottom two
ENDIF
IF ( J-A < A+1-J )
    Pixel is one of the "left" two
ELSE
    Pixel is one of the "right" two
ENDIF
```



Bilinear Interpolation:

Viene determinato il valore del nuovo pixel utilizzando la media pesata dei 4 pixel vicini nella immagine originale.

Si fa prima un interpolazione lineare nella direzione di x e poi nella direzione di y.

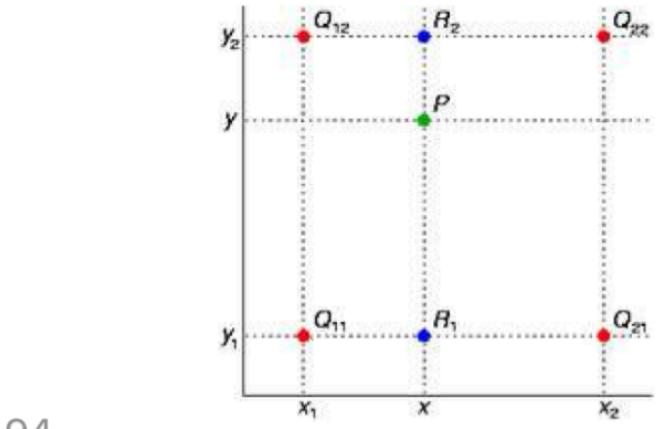
Per questo si chiama Bilineare.

$$f(R_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$

$$f(R_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})$$



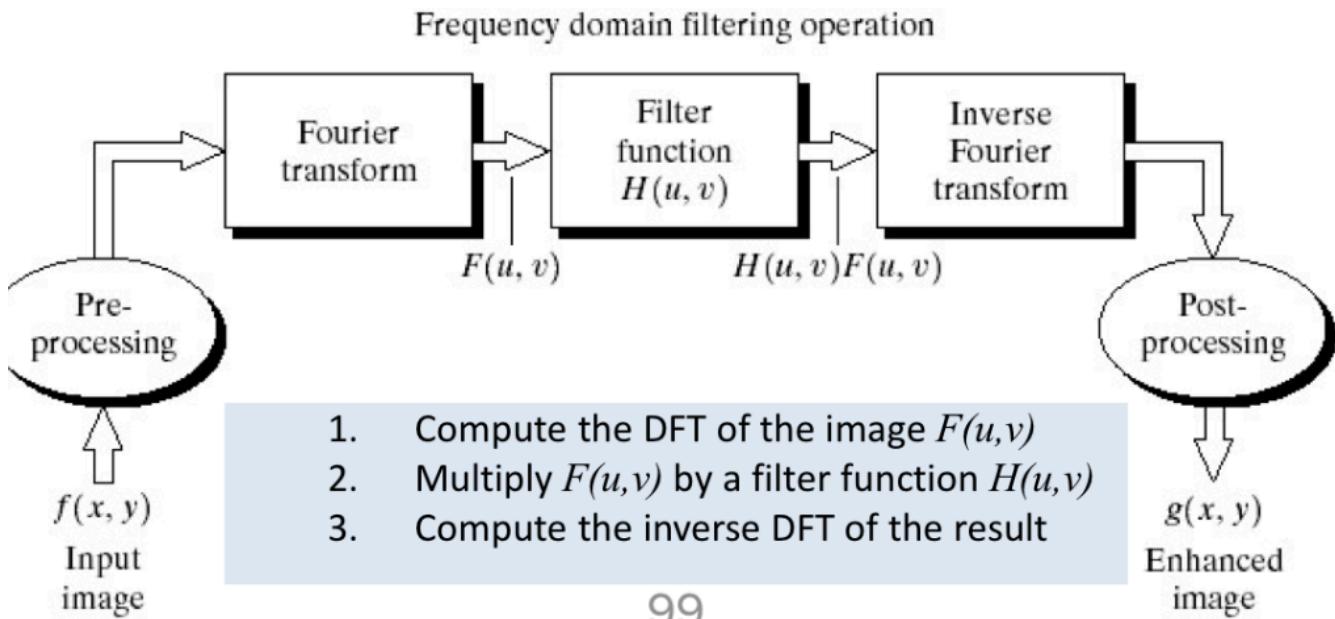
$$f(P) \approx \frac{y_2 - y}{y_2 - y_1} f(R_1) + \frac{y - y_1}{y_2 - y_1} f(R_2).$$



Bicubic Interpolation:

Mentre in quello bicubico si prendono i 16 pixels dei più vicini 4x4 vicini al pixel nell'immagine originale e il valore $f(x,y)$ vengono fatte tramite media pesata dei 16 pixel nella griglia 4x4.

Filtraggio nel dominio di frequenza:



Il filtraggio è basato su muovere il segnale dal dominio spaziale a quello di frequenza, modificare la trasformata di un immagine attraverso una funzione filtro e poi fare IDFT (ovvero l'inversa) per avere il risultato come output.

Nell'applicazione del filtro si ha la moltiplicazione **elemento per elemento** tra H e F.

$$H(u, v) \cdot F(u, v)$$

Mentre i componenti di $F(u,v)$ sono complessi filtri che usiamo tipicamente sono reali (quindi hanno fase 0)

Filtri Lowpass:

Le frequenze basse sono responsabili di far apparire alcune zone dell'area lisce mentre le frequenze alte sono responsabili dei dettagli, margini e rumori.

Un filtro che attenua le frequenze alte è chiamato **lowpass filter** altrimenti **highpass filter**.

Un'immagine che ha avuto un filtro lowpass avrà meno dettagli fini poiché il filtro ha attenuato le frequenze alte.

Mentre un filtro che è passato da un highpass avrà meno livelli di grigio nelle aree uniformi e enfatizza le zone di transizione.

Ci sono tre tipi di lowpass:

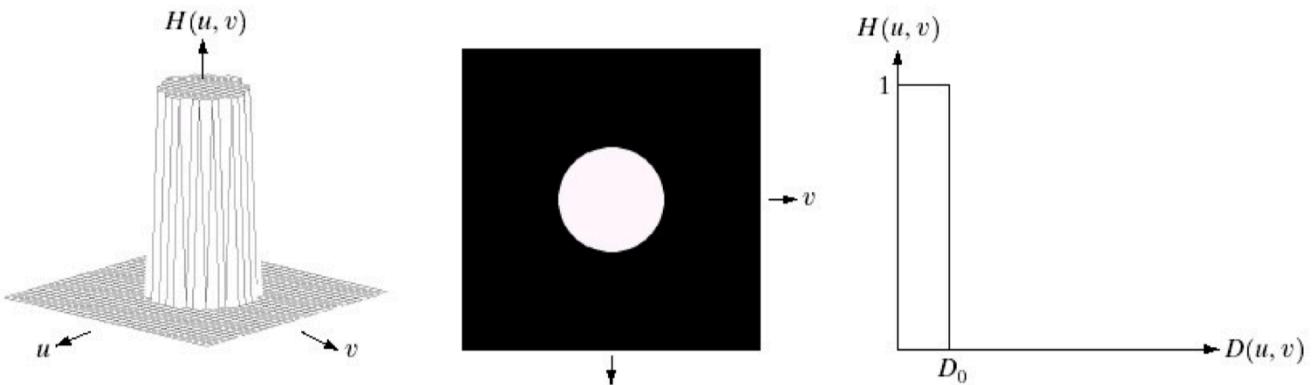
- ideale(ILPF)
- Butterworth(BLPF)
- Gaussian(GLPF)

Ideal Low Pass Filter:

Semplicemente taglia tutte le componenti con alte frequenze che sono fuori dal cerchio di raggio D_0 con centro l'origine della trasformata.

Tutte le frequenze passate sono passate senza essere attenuate mentre quelle fuori vengono totalmente attenuate.

Questo tipo di filtro non può essere fatto attraverso componenti elettriche ma **DEVE** essere computato.



Si vede, matematicamente, il filtro $H(u,v)$ come:

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

dove $D(u,v)$ è la distanza tra il punto (u,v) e il centro del rettangolo della frequenza:

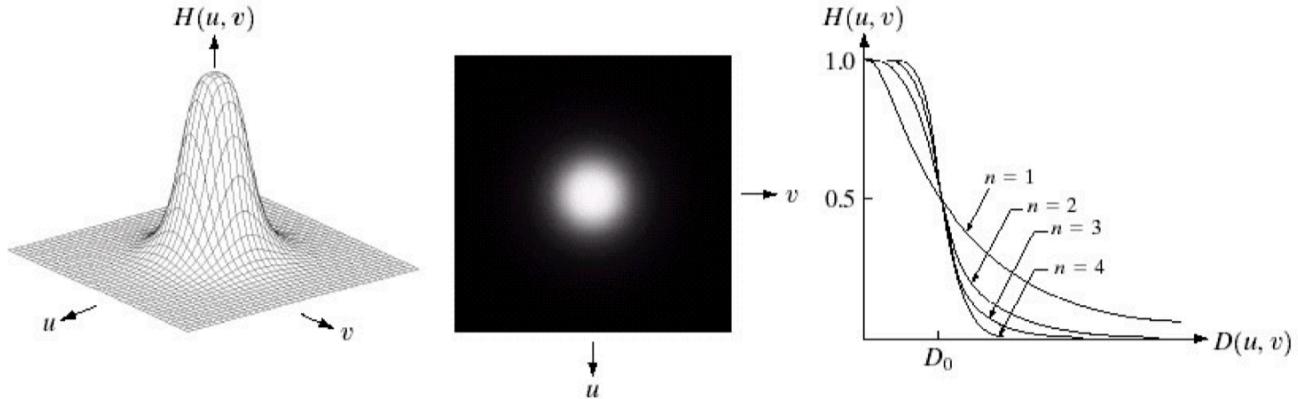
$$D(u, v) = \sqrt{(u - P/2)^2 + (v - Q/2)^2}$$

con immagine padding di dimensione $P \times Q$ e la trasformata è centrata a $(P/2, Q/2)$.

Butterworth Lowpass Filter

La funzione di trasferimento di questo filtro di ordine n con la frequenza di *cutoff* alla distanza D_0 dall'origine è questa:

$$H(u, v) = \frac{1}{1 + (D(u, v)/D_0)^{2n}}$$

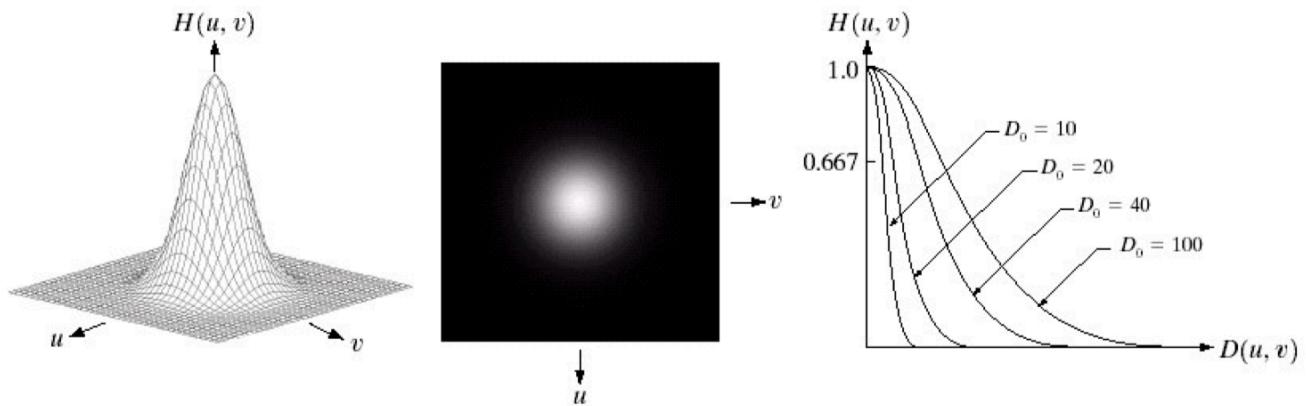


La differenza tra questi primi due filtri è la rapidità con il quale scende la funzione, infatti questa prima è graduale e non ha una discontinuità data dal *cutoff* del filtro ideale lowpass.

Gaussian Lowpass Filter

Nel filtro gaussiano invece la funzione di trasferimento è questa:

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



Questo filtro è usato per unire testo rotto nei documenti scannerizzati con poca risoluzione. Inoltre viene utilizzato per levare imperfezioni dalle foto.

Filtri Highpass:

Come si diceva all'inizio, i filtri passa alto fanno passare le frequenze alte a discapito di quelle basse, infatti sono il contrario di quelle lowpass e le possiamo vedere come:

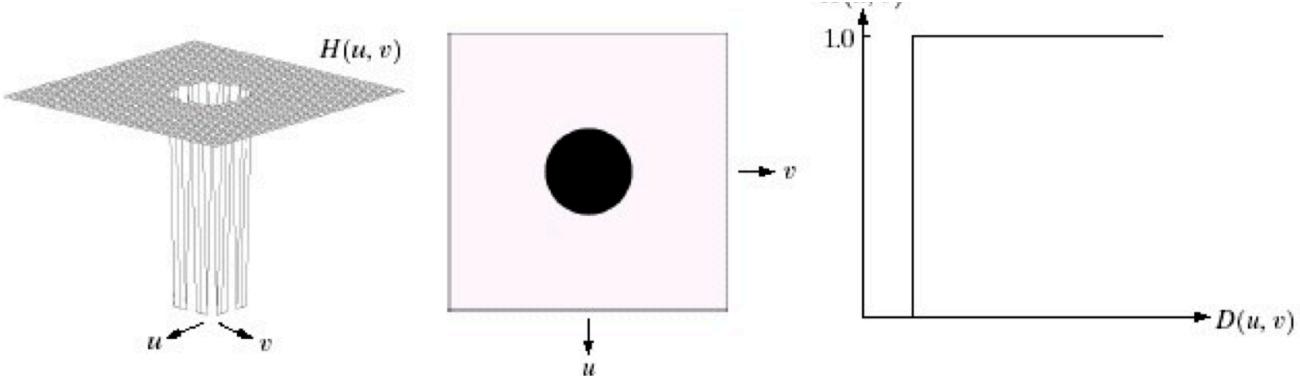
$$H_{hp}(u, v) = 1 - H_{lp}(u, v)$$

Anche qui ci saranno tre tipi di filtri:

- Ideal High Pass Filter (IHPF)
- Butterworth High Pass Filter (BHPF)
- Gaussian High Pass Filter (GHPF)

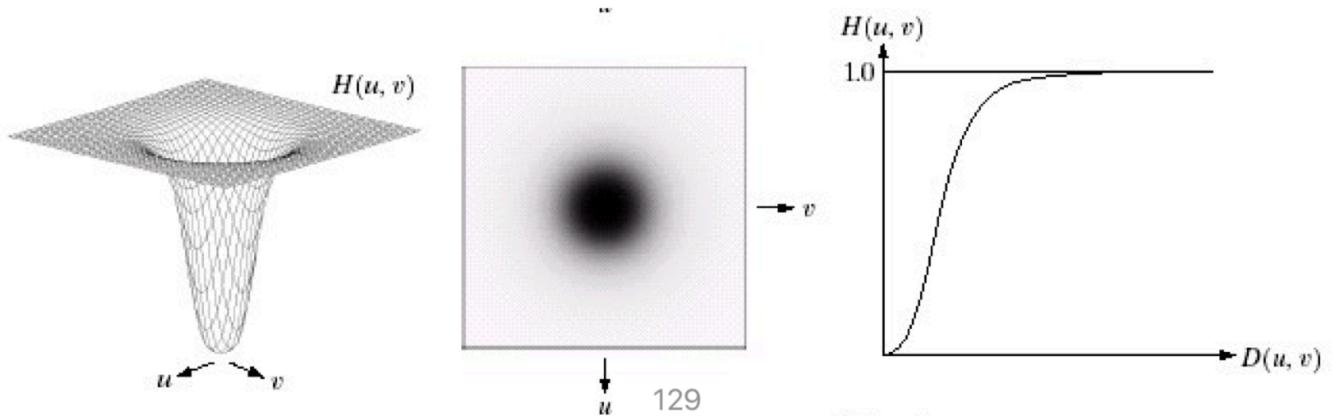
Ideal High Pass Filter

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



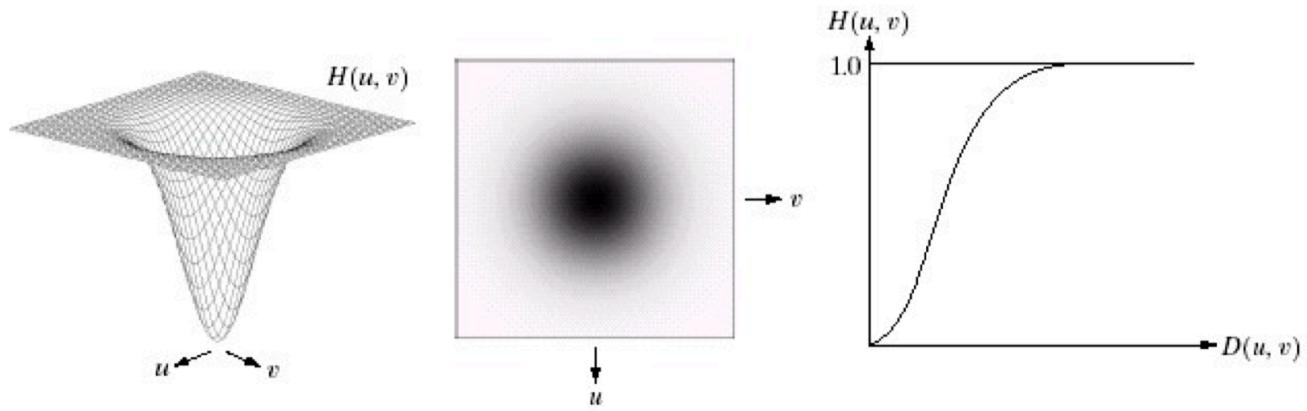
Butterworth High Pass Filter

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



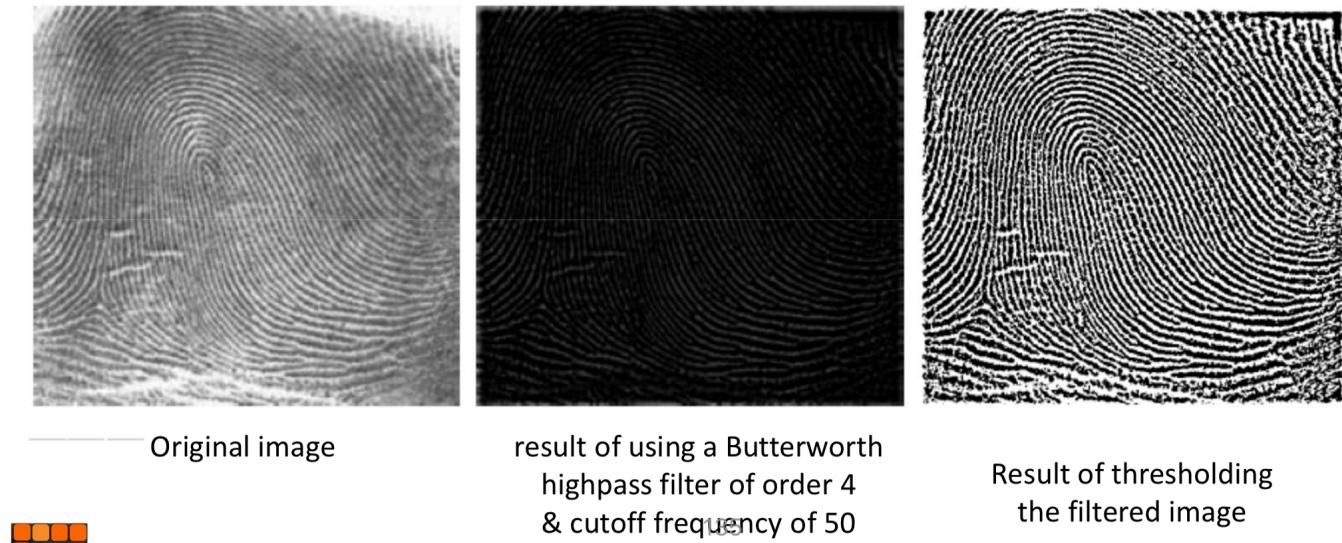
Gaussian High Pass Filter:

$$H(u, v) = 1 - e^{-\frac{D(u,v)^2}{2D_0^2}}$$



Un esempio di utilizzo di questi filtri è nel riconoscimento di un impronta digitale, migliorando le righe dell'impronta levando le macchie(componenti a frequenza bassa). Dopo di che può

essere fatto un threshold dell'immagine filtrata.



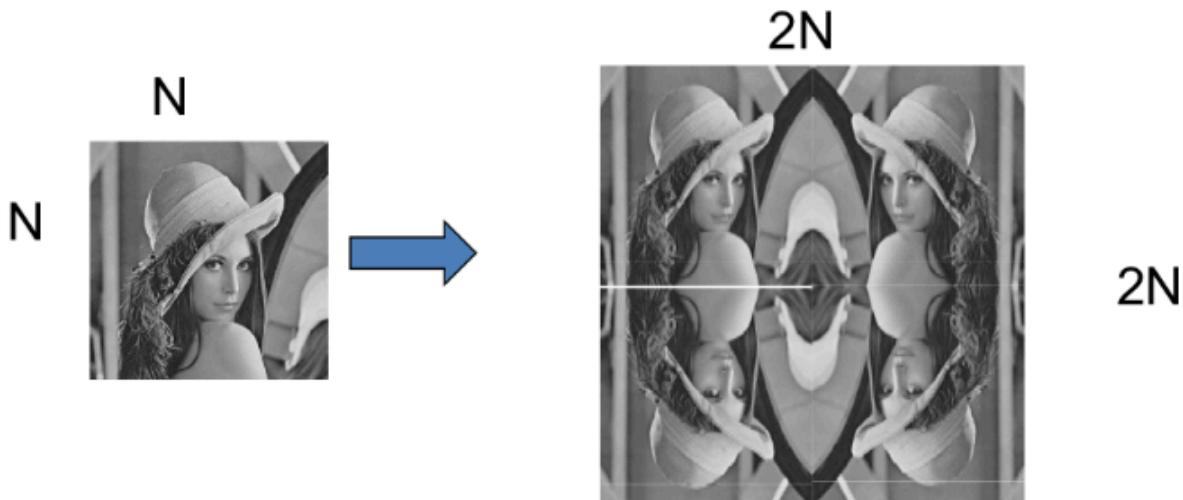
Svantaggi 2D-DFT:

Gli svantaggi della DFT sono che:

- i campionamenti di questa trasformata sono numeri complessi, quindi hanno un magnitudo(ampiezza) e una fase.
- Poichè dalla definizione di DFT l'immagine $f(x,y)$ viene rappresentata come una ripetizione di immagini, il suo spettro contiene componenti orizzontali e verticali che non corrispondono a nessun contenuto dell'immagine.
- Il segnale converge in modo lento: è difficile rappresentare variazioni forti di livello di grigio dovuto alle funzioni elementari **appiane** (scarsa compattazione energetica)

Discrete Cosine Trasform (2D-DCT)

Per evitare questi svantaggi si può definire un'altra trasformata che ha un approccio diverso alla ripetizione spaziale.



Con questa trasformata si ha una simmetrica tale che:

$$f(x, y) = f(-x, y) = f(x, -y) = f(-x, -y)$$

E se vediamo la trasformata di Fourier rendendo conto di questa caratteristica:

$$F(u, v) = \sum_{x=0}^{2M-1} \sum_{y=0}^{2N-1} f(x, y) e^{-j2\pi(\frac{ux}{M} + \frac{vy}{N})} = \sum_{x=0}^{2M-1} \sum_{y=0}^{2N-1} f(-x, y) e^{-j2\pi(\frac{-ux}{M} + \frac{vy}{N})}$$

La sommatoria fa sì che elimini le parti immaginarie e quindi si ha fase uguale a zero. Data la simmetria anche i $4N^2$ campionamenti dello spettro sono simmetrici rispetto ad origine e asse.

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

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

where $c(x) = \begin{cases} 1/\sqrt{2} & \text{if } x = 0 \\ 1 & \text{otherwise} \end{cases}$

Questa trasformata permette di eliminare le frequenze verticali e orizzontali dovute alle discontinuità spaziali e tutti i coefficienti sono reali.

Impatto energetico: è l'abilità di impacchettare l'energia di un segnale spaziale in meno coefficienti di frequenza possibili, che è molto importante per la compressione.

Basi dell'immagine DCT:

L'immagine è decomposta in un insieme di immagini chiamate **immagini base**, che quando sommate insieme ricompongono la base.

La base con N=8

$$b(x, y, u, v) = \frac{1}{4} c(u) c(v) \cdot \cos\left(\frac{\pi}{8}[u(x + 1/2)]\right) \cdot \cos\left(\frac{\pi}{8}[v(y + 1/2)]\right)$$

per ogni valore differente di u,v otteniamo un immagine spaziale di 8×8 :

- $u=v=0$: Constant image

$$b(x,y,0,0) = \frac{1}{4\sqrt{2}\sqrt{2}} \cos\left(\frac{\pi}{8} \left[0\left(x+\frac{1}{2}\right)\right]\right) \cos\left(\frac{\pi}{8} \left[0\left(y+\frac{1}{2}\right)\right]\right) = \frac{1}{8} = 0.125$$

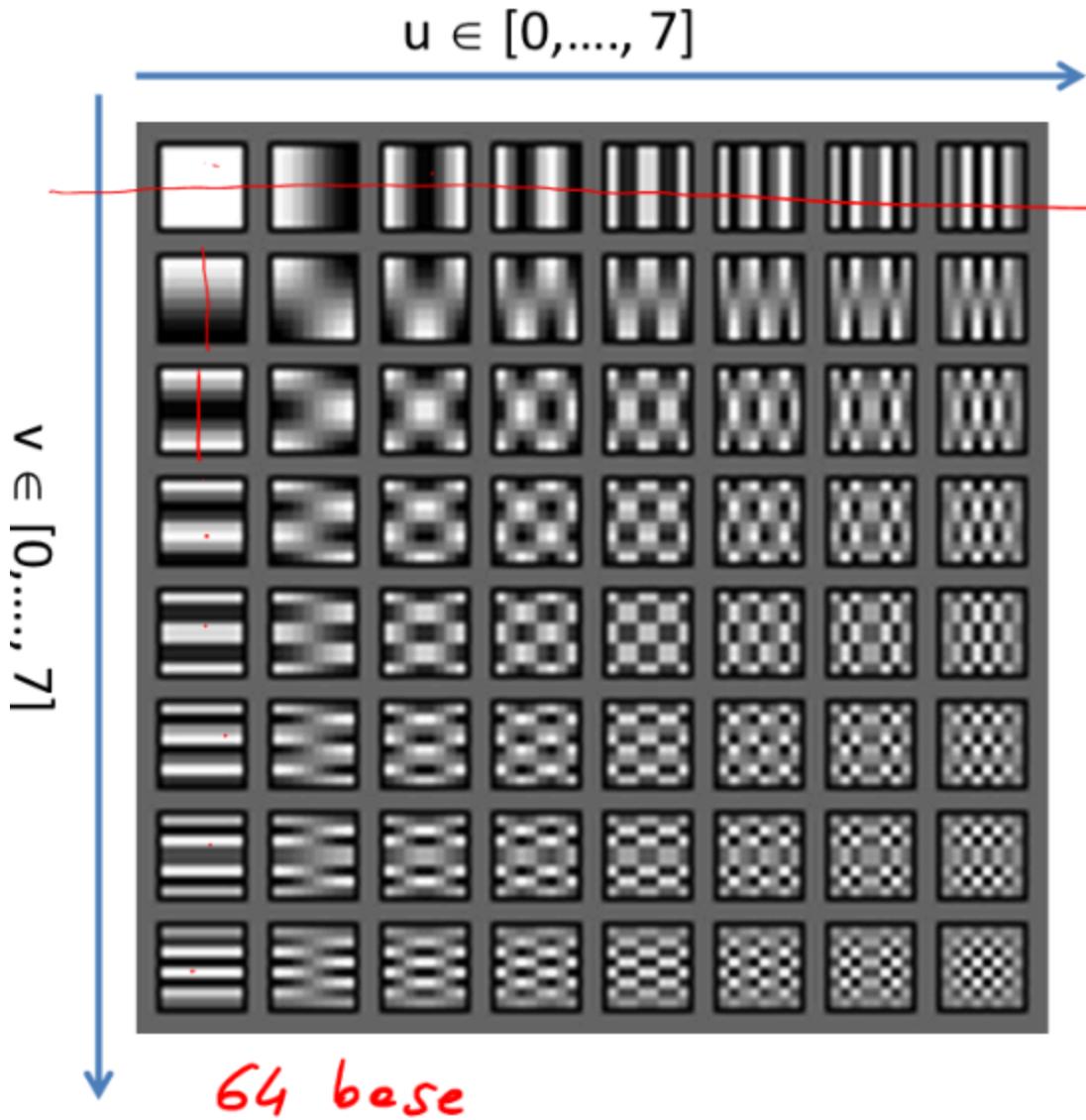
$$b(x,y,1,0) = \frac{1}{4\sqrt{2}} \cos\left(\frac{\pi}{8} \left[1\left(x+\frac{1}{2}\right)\right]\right) \cos\left(\frac{\pi}{8} \left[0\left(y+\frac{1}{2}\right)\right]\right) = \frac{1}{4\sqrt{2}} \cos\left(\frac{\pi}{8} \left[x+\frac{1}{2}\right]\right)$$

$\cancel{x=0}$ $\frac{1}{4\sqrt{2}} \cos\left(\frac{\pi}{8} \cdot \frac{1}{2}\right)$

- $u=1, v=0$: \cos along the x axis with frequency 0.5 Hz \Leftrightarrow period = 16 pixels

0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125
0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125
0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125
0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125
0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125
0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125
0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125
0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125
0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125	0,125

0,17	0,15	0,10	0,03	-0,03	-0,10	-0,15	-0,17
0,17	0,15	0,10	0,03	-0,03	-0,10	-0,15	-0,17
0,17	0,15	0,10	0,03	-0,03	-0,10	-0,15	-0,17
0,17	0,15	0,10	0,03	-0,03	-0,10	-0,15	-0,17
0,17	0,15	0,10	0,03	-0,03	-0,10	-0,15	-0,17
0,17	0,15	0,10	0,03	-0,03	-0,10	-0,15	-0,17
0,17	0,15	0,10	0,03	-0,03	-0,10	-0,15	-0,17
0,17	0,15	0,10	0,03	-0,03	-0,10	-0,15	-0,17



Quindi:

$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u, v) \cdot b(u, v, x, y)$$

Questa equazione spiega che un immagine $N \times N$ può essere ottenuta come la somma pesata di N^2 immagini basi della stessa dimensione $N \times N$ dove i pesi sono dati dai coefficienti DCT $C(u, v)$.

È meglio considerare un immagine $N \times N$ come la composizione di 8×8 riquadri

$$f_i(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C_i(u, v) \cdot b(u, v, x, y)$$

Quindi sostanzialmente viene calcolato per ogni riquadro 8×8 immagini DCT e annessi coefficienti DCT.