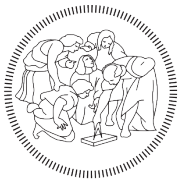

POLITECNICO
MILANO 1863

Esercizio 3 Homework 2 - Laboratrio di elaborazione di bioimmagini

Autore: Francesco Benedetto 10663326

Anno Accademico 2024/2025

Indice

1 Introduzione **1**

2 Metodo **1**

2.1 Caricamento e visualizzazione dell'immagine originale 1

2.2 Analisi istogramma 2

2.3 Identificazione Rumore Sale e Pepe . . . 2

2.4 Applicazione filtro mediano 2

2.5 Risultati post-filtro mediano 3x3 2

2.6 Analisi dettagliata rumore sale e pepe . 3

2.7 Rimozione rumore sinusoidale 4

2.8 Enhancement finale 5

3 Risultati **5**

4 Conclusioni **5**

1 Introduzione

Questo report descrive l'analisi e la rimozione del rumore da un'immagine corrotta. L'obiettivo dell'esercizio è identificare il tipo di rumore presente nell'immagine, stimarne i parametri numerici e progettare una strategia per la sua rimozione.

2 Metodo

L'approccio adottato nel codice è suddiviso in vari passaggi. Ogni passaggio ha lo scopo di preparare l'immagine per una corretta elaborazione e miglioramento. Di seguito sono descritti i passaggi fondamentali.

2.1 Caricamento e visualizzazione dell'immagine originale

Il primo passaggio consiste nel caricare l'immagine di input in formato .mat, che rappresenta una radiografia di due ginocchia. L'immagine di input ha dimensioni di 1024×1024 pixel con valori di intensità di tipo `uint8` (valori compresi tra 0 e 255).

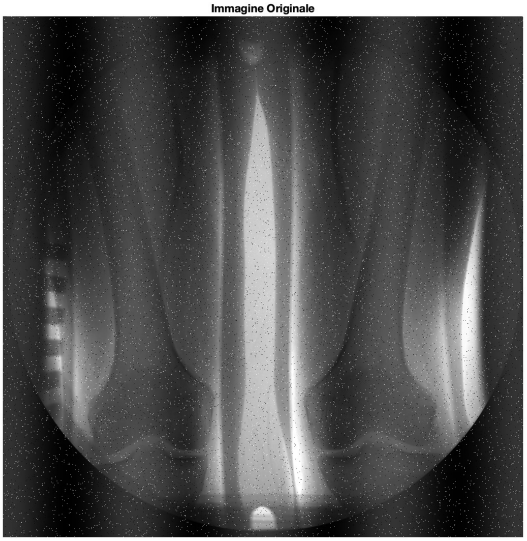

Figura 1: Immagine iniziale

2.2 Analisi istogramma

L'analisi dell'*istogramma* evidenzia tre caratteristiche principali: la presenza di due picchi pronunciati ai valori di intensità **0** e **255**, indicativi di componenti estreme nell'immagine, e una distribuzione a forma di campana centrata intorno al valore **52**, che rappresenta l'intensità più frequente.

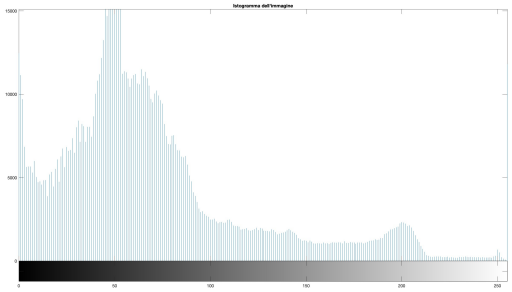

Figura 2: Istogramma immagine corrotta

2.3 Identificazione Rumore Sale e Pepe

Per ottenere una rappresentazione chiara della variazione dell'intensità nell'immagine, è stato scelto di analizzare il profilo di intensità della riga centrale, la quale copre un ampio intervallo di valori. Dall'osservazione di questo profilo, emerge chiaramente che l'immagine è affetta da rumore, caratterizzato dalla presenza di *spike* positivi e negativi, distribuiti in modo casuale. Questo fenomeno è indicativo della presenza di rumore di tipo *sale e pepe*. Per supportare questa ipotesi e dimostrare che il rumore non è confinato solo alla riga centrale, è stata inoltre visualizzata l'intera immagine in uno spazio tridimensionale, dove la terza dimensione rappresenta l'intensità.

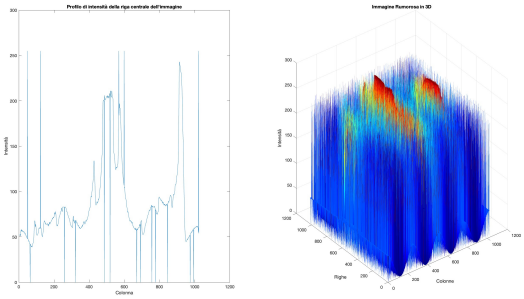

Figura 3: Da queste due immagini è possibile identificare gli *spike* che raggiungono valore 0 (nero - *pepe*) e valore 255 (bianco - *sale*)

2.4 Applicazione filtro mediano

Essendo stato identificato un rumore di tipo *sale e pepe*, è stato scelto di applicare un filtro mediano per la sua rimozione. Sono state esplorate diverse dimensioni della finestra del filtro, calcolando per ciascuna di esse le metriche *SNR* (Signal-to-Noise Ratio) e *PSNR* (Peak Signal-to-Noise Ratio). Alla fine, è stata selezionata la dimensione della finestra (3x3) che ha prodotto i valori più elevati di *SNR* e *PSNR*. Si noti che, per il calcolo delle metriche, l'immagine filtrata è stata considerata come stima del segnale pulito.

Le formule utilizzate per il calcolo di *SNR* e *PSNR* sono le seguenti:

$$\text{SNR} = 10 \cdot \log_{10}\left(\frac{\text{signal\_power}}{\text{noise\_power}}\right)$$

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{255^2}{\text{MSE}}\right)$$

2.5 Risultati post-filtro mediano 3x3

Dopo aver applicato il filtro mediano con finestra 3×3, si osserva che il rumore di tipo *sale e pepe* è stato efficacemente rimosso dall'immagine. Tale risultato è confermato dal profilo di intensità della riga centrale nell'immagine filtrata, nonché dalla distribuzione 3D dell'immagine stessa, che appare significativamente più *smooth* rispetto alla versione rumorosa.

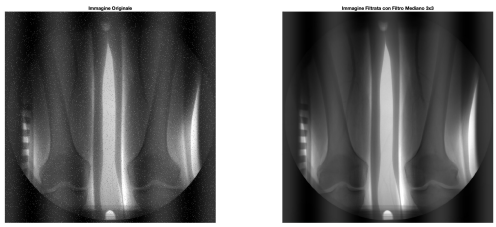

Figura 4: Pre e post filtro mediano

Una volta eliminato il rumore *sale e pepe*, è stato possibile identificare la presenza di un ulteriore componente rumoroso. Questo nuovo rumore si presenta sotto forma di una sinusoide 2D che varia esclusivamente lungo l'asse x. La sinusoide ha un'ampiezza massima pari a 26 (circa la metà del picco dell'istogramma, che è 52) e una frequenza di 4/1024, con fase nulla. Tale componente è visibile facilmente nella mesh dell'immagine post-filtraggio mediano. Essa infatti mostra una base sinusoidale con 4 cicli su 1024 campioni che parte da 0 e arriva a 52.

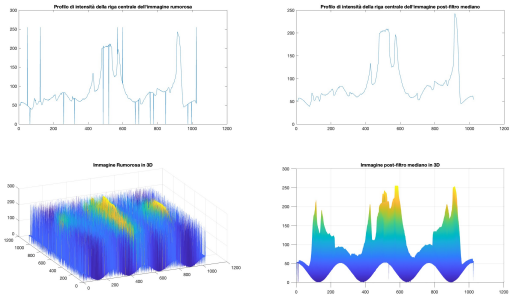

Figura 5: Pre e post filtro mediano. La mesh dell'immagine a destra è stata orientata in modo di essere perpendicolari all'asse x e vedere come varia l'intensità lungo le righe, potendo identificare la sinusoide.

2.6 Analisi dettagliata rumore sale e pepe

Per determinare il parametro di densità del rumore d utilizzato nell'aggiunta del rumore *sale e pepe* all'immagine, è stato eseguito il seguente processo:

- Sono stati conteggiati i pixel neri (valore 0) nell'immagine rumorosa e nell'immagine filtrata, per determinare la quantità di pixel *pepper* (neri) aggiunti dal rumore. La differenza tra i pixel neri nell'immagine rumorosa e in quella filtrata ha fornito il numero di pixel *pepper*.
- In modo simile, sono stati conteggiati i pixel bianchi (valore 255) nell'immagine rumorosa e in quella filtrata, per determinare il numero di pixel *salt* (bianchi) aggiunti dal rumore. La differenza tra i pixel bianchi nell'immagine rumorosa e in quella filtrata ha dato il numero di pixel *salt*.
- Il numero totale di pixel alterati, ovvero la somma dei pixel *salt* e *pepper*, è stato calcolato.
- Infine, la densità del rumore d è stata ottenuta come il rapporto tra il numero totale di pixel alterati e il numero totale di pixel nell'immagine. Formalmente, la densità del rumore d è definita come:

$$d = \frac{\text{num\_salt\_pixels} + \text{num\_pepper\_pixels}}{\text{total\_pixels}}$$

dove `num_salt_pixels` è il numero di pixel *salt*, `num_pepper_pixels` è il numero di pixel *pepper*, e `total_pixels` è il numero complessivo di pixel nell'immagine.

Il calcolo della densità del rumore ha fornito i seguenti risultati:

- Numero di pixel *pepper* (neri): 10560
- Numero di pixel *salt* (bianchi): 10420
- Densità totale del rumore d: 0.020008

Questi valori sono stati utilizzati per caratterizzare l'intensità del rumore presente nell'immagine.

Infine, per visualizzare la distribuzione del rumore, sono state generate le mappe che evidenziano la posizione dei pixel *salt* e *pepper* nelle immagini rumorosa e filtrata, come mostrato nelle seguenti immagini.

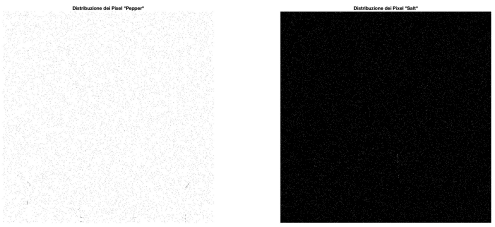

Figura 6: Distribuzione dei pixel (sale e pepe) che sono cambiati pre e post filtro mediano

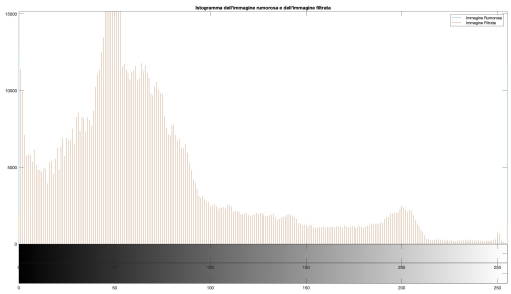

Figura 7: Istogrammi pre e post filtro mediano. Si pùò notare come siano variati in numero solo i pixel bianchi/neri

2.7 Rimozione rumore sinusoidale

Per eliminare il rumore sinusoidale presente nell'immagine filtrata, è stato necessario prima costruire un modello della sinusoide stessa. La sinusoide in questione è un rumore additivo che varia lungo l'asse x e ha le seguenti caratteristiche:

- *Ampiezza*: La metà del picco dell'istogramma, che corrisponde a 26. Questo valore è stato normalizzato nel range [0, 1] dividendo per 255, ottenendo un'ampiezza di $\frac{26}{255}$.
- *Frequenza*: La sinusoide ha una frequenza pari a $\frac{4}{1024}$, il che implica che la sinusoide completa fa 4 cicli in un'immagine di 1024 pixel.
- *Fase*: La fase iniziale è stata impostata a 0, senza alcuno spostamento rispetto all'origine.
- *Theta*: La sinusoide varia solo lungo x per questo motivo l'orientamento rispetto all'asse x (*Theta*) è uguale a 0.
- *Offset*: È stato aggiunto un offset per assicurarsi che i valori della sinusoide siano compresi nel range [0, 52], in modo simile ai valori della mesh a destra in *Figure 5*. Se non fosse stato applicato questo bias la sinusoide avrebbe avuto valori nel range [-26, 26].

La sinusoide è stata quindi costruita utilizzando la funzione `immcos`, che genera un'onda coseno con ampiezza, frequenza e fase definiti sopra. L'offset è stato aggiunto per ottenere valori positivi che vanno da 0 a 52, come nella distribuzione del rumore sinusoidale osservata nell'immagine.

Matematicamente, la sinusoide $W_A(x)$ è descritta come segue:

$$W_A(x) = \text{amp} \cdot \cos(2\pi F \cdot x + \phi)$$

dove:

- `amp` è l'ampiezza normalizzata,
- F è la frequenza della sinusoide,
- x è la variabile spaziale lungo l'asse orizzontale dell'immagine,
- ϕ è la fase iniziale, che in questo caso è 0.

Per garantire che la sinusoide rientrasse nel range appropriato, è stato aggiunto l'offset $\min(W_A)$ all'immagine sinusoidale.

Successivamente, per rimuovere il rumore sinusoidale dall'immagine filtrata, è stata sottratta l'immagine sinusoidale W_A dall'immagine filtrata I_{filtered}. Essendo il rumore sinusoidale di tipo additivo, questa sottrazione ha permesso di ottenere un'immagine "pulita", priva della componente sinusoidale.

L'operazione di sottrazione è stata eseguita utilizzando la funzione `imsubtract` di MATLAB:

$$\text{diff} = I_{\text{filtered}} - W_A$$

Dove `diff` rappresenta l'immagine risultante, che contiene solo la parte dell'immagine che non è stata influenzata dal rumore sinusoidale.

Infine, per visualizzare l'efficacia della sottrazione, sono stati mostrati tre grafici:

- Il primo mostra l'immagine della sinusoide 2D W_A.
- Il secondo mostra l'immagine filtrata I_{filtered}.
- Il terzo mostra la differenza tra l'immagine filtrata e la sinusoide, che rappresenta l'immagine "pulita".

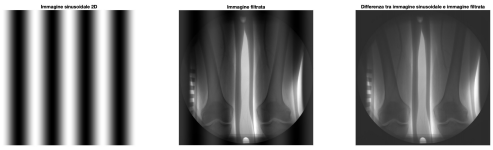

Figura 8: Eliminazione sinusoide

Questi grafici confermano che la sottrazione della sinusoide ha rimosso con successo il rumore sinusoidale dall'immagine.

2.8 Enhancement finale

Per analizzare i risultati ottenuti dopo la sottrazione della sinusoide, è stato visualizzato l'istogramma dell'immagine risultante, `diff`. Come evidenziato dal grafico, il picco corrispondente al valore 52, che rappresentava il rumore sinusoidale, è stato efficacemente rimosso.

Successivamente, per migliorare ulteriormente la visibilità dei dettagli nell'immagine, è stato applicato un processo di *contrast stretching* tramite la funzione `imadjust`. In particolare, è stata utilizzata la funzione `stretchlim` per determinare automaticamente i limiti minimi e massimi dell'immagine, che sono poi stati estesi all'intervallo completo [0, 1] per ottimizzare la distribuzione dei valori di intensità.

3 Risultati

Parametro	**Valore**
Rumore Sale e Pepe	
Numero di pixel "pepper" (neri)	10,560
Numero di pixel "salt" (bianchi)	10,420
Densità totale del rumore d	0.020008
Componente Sinusoidale	
Ampiezza	26
Dimensione Immagine	1024
Theta	0
Frequenza	4 cicli su 1024 pixel
Fase	0

Tabella 1: Risultati relativi al rumore sale e pepe e alla componente sinusoidale.

4 Conclusioni

Dopo aver identificato le tipologie di rumori aggiunte all'immagine e i rispettivi parametri, è stato efficacemente restaurata l'immagine originale.

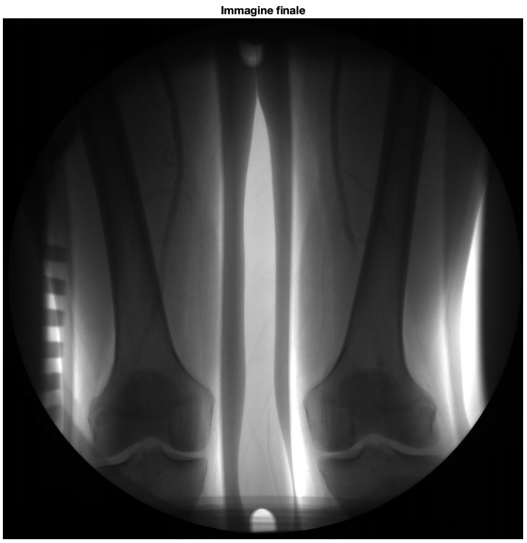

Figura 9: Immagine finale