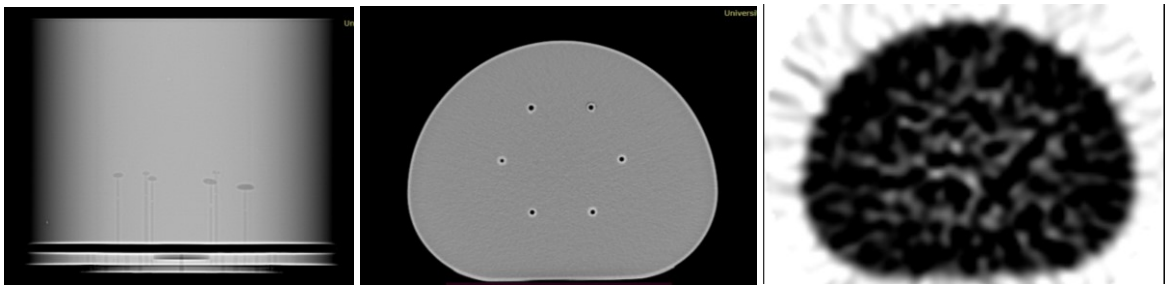
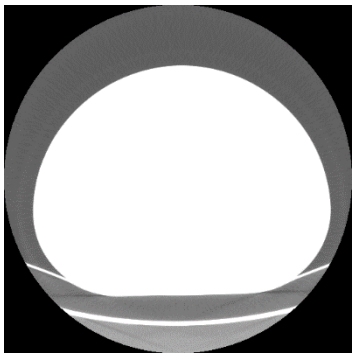
ESERCITAZIONE 2 (FILTRAGGIO, 16/03/2022)

Scopo dell'esercitazione è realizzare un programma che consenta di valutare vari algoritmi di filtraggio 3D su di un fantoccio CT in termini di CNR e conservazione delle transizioni. Il fantoccio che utilizziamo è un fantoccio TAC-PET, usato per il controllo di qualità in questo tipo di macchine ibride (https://wiki.cancerimagingarchive.net/display/Public/RIDER+Phantom+PET-CT).



Nella directory PHANTOM_CT_PET troviamo tre serie, un localizer (SCOUT) che serve come guida per le acquisizioni successive, una immagine 3D CT (serie 2) ed una immagine PET (serie 401). Nell'esercitazione utilizziamo l'immagine CT, composta da 63 slices.

Come si osserva attraverso una opportuna operazione di windowing con un visualizzatore DICOM, nell'immagine TAC è presente un'area non ricostruita (zero padding) con valore convenzionale -3024 ed un area a segnale nullo con valore intorno a -1000 (numero HU o di Hounsfield dell'aria). Il fantoccio è caratterizzato da un valore HU intorno a 60. L'immagine è quindi espressa in HU e contiene quindi valori negativi.



Sarebbe possibile memorizzare tali valori nel DICOM utilizzando un formato "signed 16-bit integer", che permette di codificare interi a 15 bit positivi e negativi. Tuttavia in questo caso si è preferito memorizzare "unsigned 16 bit integer" (interi positivi) ed utilizzare due campi del DICOM per codificare i valori corretti. Tali campi sono:

0028,1052 Rescale Intercept

0028,1053 Rescale Slope

Se questi campi sono presenti, una libreria DICOM correttamente configurata leggerà i valori interi positivi dell'immagine I e calcolerà i valori corretti Ic come:

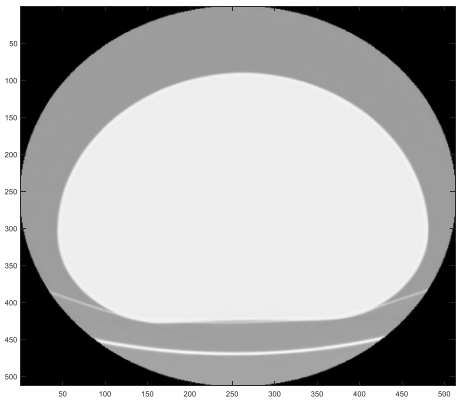
Ic = RescaleIntercept + RescaleSlope*I

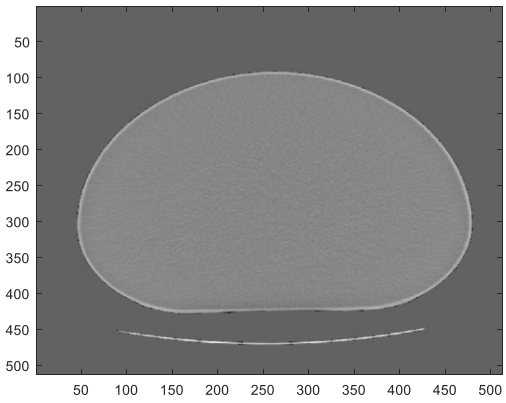
In pratica viene definita una funzione lineare che implementa una hash table che converte valori interi positivi a 16 bit (quelli del DICOM) in un qualsiasi range di uscita. Il campo RescaleType = HU ci dice che lo scopo è quello di ottenere una immagine in scala HU.

La funzione dicomread del MATLAB ignora erroneamente questi campi (almeno fino alla versione 2021b), quindi sarà preliminarmente necessario implementare la conversione corretta utilizzando i campi prima indicati in modo da ottenere un array contenente i valori HU corretti. Sarà anche opportuno trasformare il format odei dati da int16 a double per le seguenti elaborazioni.

Poiché vogliamo applicare un filtraggio 3D, il primo passo è quello di verificare se il volume di dati è isotropo leggendo gli opportuni campi DICOM e nel caso non lo sia operare una interpolazione trilineare per ottenere un volume isotropo. Si possono utilizzare le funzioni ***interp3d/meshgrid*** o ***imresize3*** con le opportune opzioni. Il volume ottenuto può essere visualizzato con **volumeViewer** (Matlab >= R2017a).

Una volta ottenuto il volume interpolato, vogliamo sperimentare vari tipi di filtraggi in 3D e valutare l'SNR e la conservazione delle transizioni sul cilindro a massimo segnale. Per semplicità calcoleremo i parametri su una singola slice in 2D (quella centrale) attraverso la definizione di una opportuna ROI. Quindi eseguiamo un filtraggio 3D ma misuriamo l'SNR in 2D su una singola slice. Nella slice centrale il fantoccio è omogeneo, quindi possiamo definire una ROI abbastanza grande da poter valutare in modo corretto le due componenti dell'SNR. Notiamo che il fantoccio è racchiuso in una struttura con un piccolo spessore di segnale superiore al fantoccio stesso invisibile nella scala di grigi di default del Matlab ma che si vede se facciamo una operazione di windowing utilizzando i campi in WindowCenter e WindowWidth.





Quello che vogliamo fare è calcolare il valore di SNR delle immagini non elaborate nella ROI è poi confrontare questo valore con quello ottenuto da tre metodi di filtraggio nella stessa ROI. Gli algoritmi di filtraggio da sperimentare includono:

1) Un filtro a media mobile con kernel 7x7x7

2) Un filtro gaussiano con kernel 7x7x7 e valore di sigma ottimizzato per massimizzare l'SNR e conservare le transizioni.

3) Un filtro adattivo di wiener con kernel 7x7x7

I primi due filtri sono convolutivi e possono essere implementati con le funzioni ***fspecial3*** (MATLAB>=R2018b) e ***imfilter***, o con la funzione ***conv3*** definendo gli opportuni kernel. Per il filtro Gaussiano è anche disponibile la funzione ***imgaussfilt3*** (MATLAB>=R2018b) che opera direttamente il filtraggio senza bisogno di utilizzare fspecial.

Il filtro di wiener bidimensionale è implementato dalla funzione ***wiener2***, purtroppo non esiste la versione 3D che va quindi implementata, tenendo conto della formulazione del filtro di Wiener:

$$I_W = I_{MM} + \frac{(I_{VAR} - \sigma^2)}{I_{VAR}}(I_{OR} - I_{MM})$$

Nella formula riconosciamo l'immagine 3D originale I_{OR}, l'immagine filtrata a media mobile I_{MM} che avremo calcolato al passo precedente, σ che è la deviazione standard del rumore stimato sull'immagine, e I_{VAR} che è la mappa della varianza (σ^2) sull'immagine 3D.

Dal punto di vista implementativo, se I_{VAR} >> σ^2 il filtro non esegue nessun filtraggio (contorni), mentre se I_{VAR} = σ^2 viene eseguito un filtraggio a media mobile. Se I_{VAR} < σ^2 il contributo del secondo termine diventa negativo e il funzionamento del filtro non è ben definito. Essendo la condizione I_{VAR} < σ^2 corrispondente ad un tessuto omogeneo vogliamo che in questa condizione venga eseguito un filtraggio a media mobile. L'implementazione del filtro diventa quindi:

$$I_W = I_{MM} + \alpha(I_{OR} - I_{MM})$$

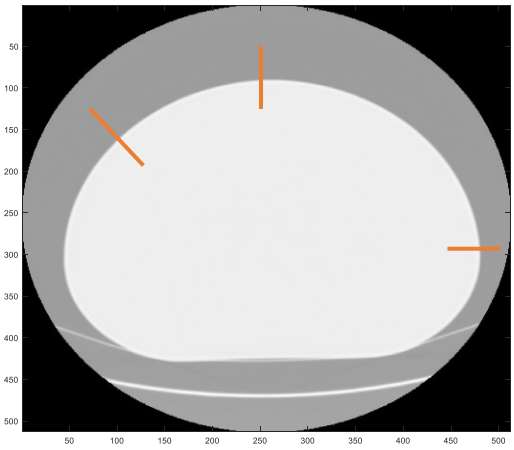
con

$$\alpha = \begin{cases} \frac{(I_{VAR} - \sigma^2)}{I_{VAR}} & se\ I_{VAR} \geq \sigma^2 \\ 0 & altrove \end{cases}$$

Per il calcolo di I_{VAR}, MATLAB include la funzione ***stdfilt*** che funziona solo su immagini 2D quindi è necessario implementarne la versione 3D per un kernel di dimensioni uguali a quello con cui si esegue il filtraggio. Per gli scopi dell'esercitazione si può ignorare il problema dei bordi. Il valore di σ andrà valutato correttamente come visto a lezione in modo da ottenere il corretto funzionamento del filtro. Ovviamente il valore di σ che ci interessa è quello del fantoccio dove facciamo la misura di SNR.

Si nota dalla formula che il calcolo diverge per valori di I_{VAR} nulli (regioni di zero padding). Tali regioni vanno quindi escluse dal filtraggio se presenti oppure a I_{VAR} va sommata una piccola quantità ε che evita il blocco del programma.

Per quanto riguarda la conservazione delle transizioni, possiamo estrarre lo stesso profilo dall'immagine originale e dalle tre immagini filtrate che misuri l'acutezza della transizione tra il fantoccio e lo sfondo. Il profilo deve essere il più possibile perpendicolare al fantoccio come negli esempi in figura.



In conclusione, il risultato dell'esercitazione è una tabella che riporta per le quattro immagini estratte dai volumi originale e filtrati i valori di SNR e acutezza.

Per visualizzare l'effetto dei vari algoritmi, è anche utile visualizzare la differenza tra l'immagine filtrata e quella originale e confrontare i profili attraverso un plot.

Immagine	SNR fantoccio	Acutezza
Originale		
Filtrata media mobile		
Filtrata filtro Gaussiano		
Filtrata filtro di Wiener		