دانشگاه صنعتی شریف

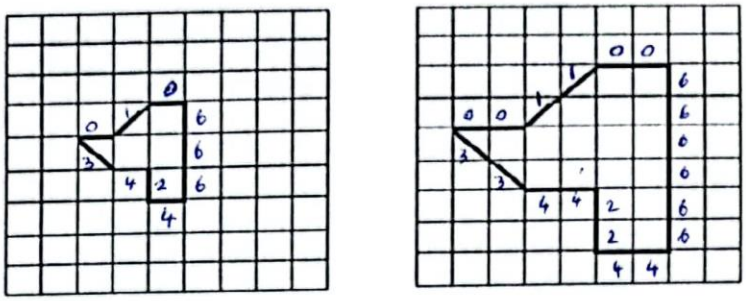
دانشکده مهندسی برق

مسعود ناطقی ۹۶۱۰۲۵۶۷

تمرین پردازش و تحلیل تصاویر پزشکی

دکتر فاطمی‌زاده

①

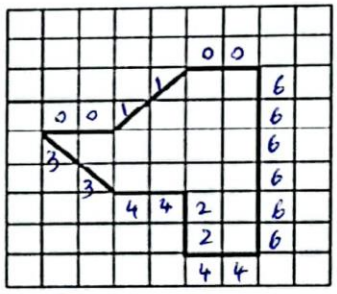


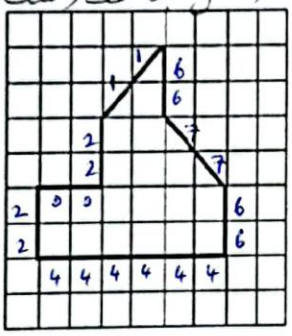
الف) بصورت واضح تغییر نقطه شروع تاثیری در کدگذار ندارد. زیرا اعدادی که با رنگ آبی مشخص شده اند (جهت ها) با تغییر نقطه شروع عوض نمی شوند و کاری که ما انجام می دهیم فقط شمردن تکرر هر یک از این اعداد است

با scale کردن اتفاق که می افتد این است که هر یک از bin های هیستوگرام در عریب scale ضرب می شوند.

بطور مثال : $h_{scale}(2) = scale \times h(2)$

حال کافی است اثر چرخش را بررسی کنیم. این کار را با استفاده از شکل بالا سمت راست انجام می دهیم.





← چرخش ۹۰° ساعتگرد

با دقت در سمت راست متوجه می شویم $h(7) = 0$. درحالیکه که در شکل سمت چپ $h(7) = 2$. بنابراین عدد حبید در هیستوگرام ظاهر شده. پس این نحوه کدگذاری وابسته به چرخش است.

با توجه به اینکه در این روش کد از scale تا حدودی مستقل است و به نقطه اولیه وابسته نیست از جمیت دارد. دقت می کنیم که با در نظر داشتن نقطه اولیه، در chain code سازده بایستی سازی برای پیدا کردن min integer code انجام شود.

ب) ابتدا کد دو شکل را می نویسیم :

Non-Convex : 0766066443542222021 0

Convex : 07765443311

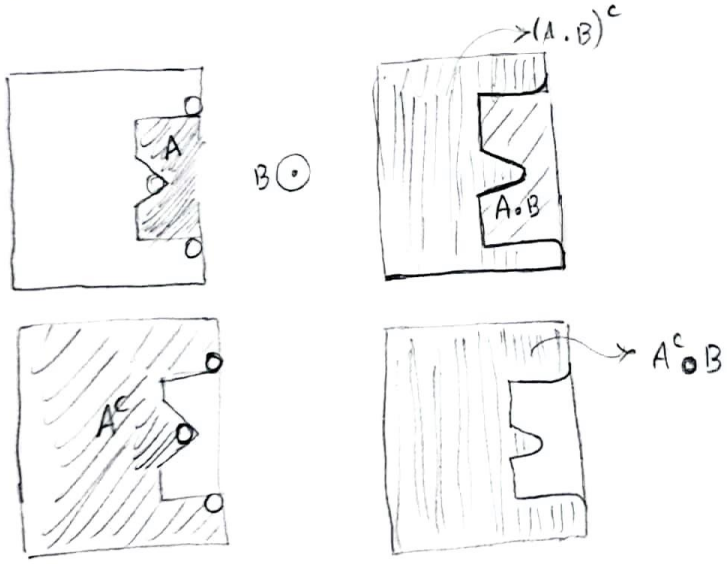
اگر رقم اول از سمت چپ را کنار بگذاریم، رقم ها بایستی قرینه باشند.

$$(A \bullet B)^c = ((A \oplus B) \ominus B)^c$$

می‌دانیم: $\begin{cases} (A \ominus B)^c = A^c \oplus \hat{B} \\ (A \oplus B)^c = A^c \ominus \hat{B} \end{cases}$

$$\rightarrow (A \bullet B)^c = (A \oplus B)^c \oplus \hat{B} = (A^c \ominus \hat{B}) \oplus \hat{B} = A^c \circ \hat{B} = A^c \circ B$$

(B متقارن)

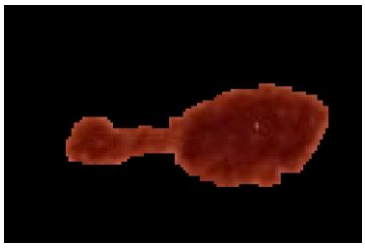


۱. الف)

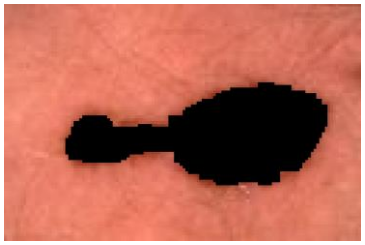
```
se1 = strel("square", 7);
se2 = strel("square", 3);
img1Enh = imdilate(img1Bin, se1);
img1Enh = imerode(img1Enh, se2);
```

final mask

hit



miss



original image

ب)

```
se1 = strel("square", 3);
img2Enh = imerode(img2Bin, se1);
img2Enh = imdilate(img2Enh, se1);
```

final mask

hit

miss

original image

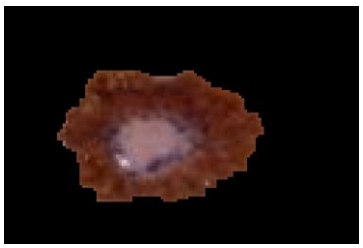
ج) عملیات و seهای استفاده شده بصورت زیر هستند:

```
se1 = strel("square", 5);
se2 = strel("square", 3);
img3Enh = imdilate(img3Bin, se1);
img3Enh = imclose(img3Enh, se1);
img3Enh = imfill(img3Enh,
"holes");
img3Enh = imerode(img3Enh, se2);
```

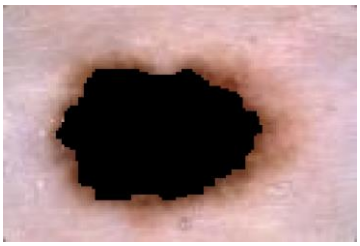
و نتیجه بصورت زیر شد:

final mask

hit



miss



original image

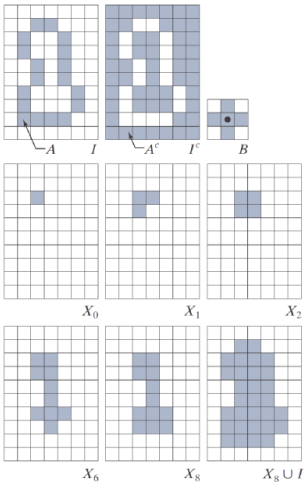
اگر از دستور imfill استفاده نمی‌کردیم می‌بایست الگوریتم زیر[1] را اجرا می‌کردیم که در آن ابتدا یک نقطه درون حفره را انتخاب می‌کنیم و الگوریتم را تا همگرا شدن ادامه می‌دهیم.

› Hole Filling Formulation:

$$X_k = (X_{k-1} \oplus B) \cap I^c$$

› Start inside the hole

› Repeat until convergence



[1] Digital Image Processing, Morphological Image Processing Slides, Dr. Emad Fatemizadeh

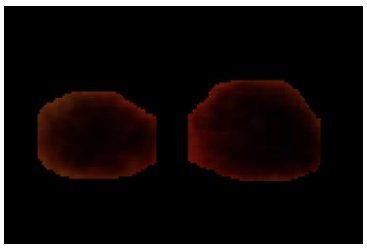
د) عملیات و seهای استفاده شده بصورت زیر هستند:

```
img4Enh = imopen(img4Bin,
strel("square", 17));
img4Enh = imerode(img4Enh,
strel("square", 10));
img4Enh = imdilate(img4Enh,
strel("square", 7));
img4Enh = imopen(img4Enh,
strel("square", 20));
```

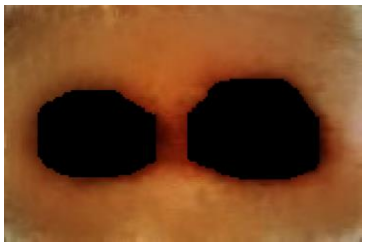
و نتیجه بصورت زیر شد:

final mask

hit



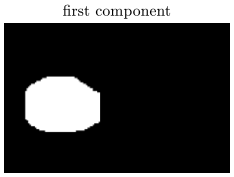
miss



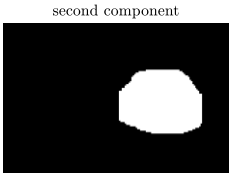
original image

هـ)

first component

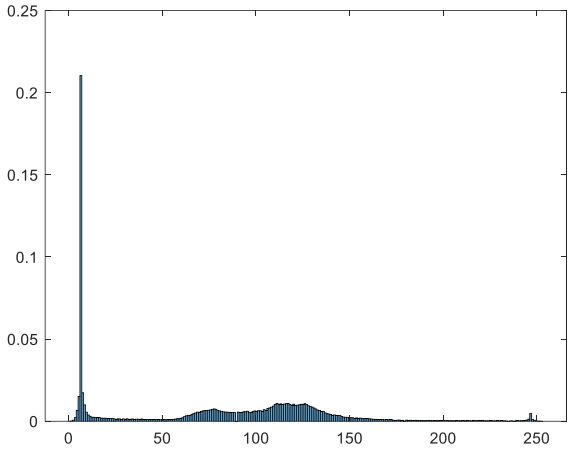


second component



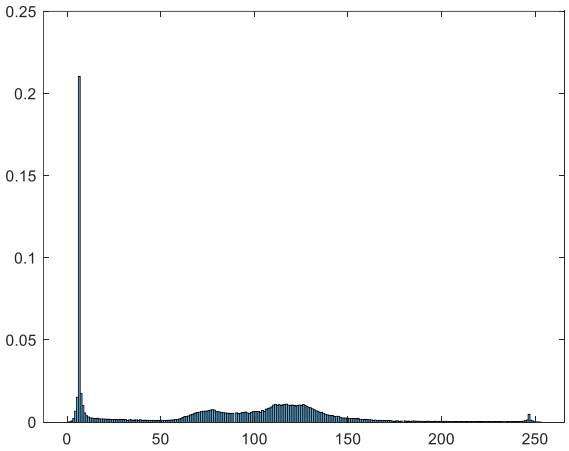
number of components = 2

۲. الف)




```
brain.jpg features
mean = 79.4222
variance = 3239.7252
uniformity = 0.049588
entropy = 6.4437
```

ب)




```
sample.png features
mean = 79.4222
variance = 3239.7252
uniformity = 0.049588
entropy = 6.4437
```

بله؛ بی‌شمار تصویر دیگر با ویژگی‌های یکسان وجود دارد.

ج)

```
glcms1 = graycomatrix(img1, "NumLevels", 256);
glcms2 = graycomatrix(img2, "NumLevels", 256);
```

ابعاد این ماتریس مربعی برابر تعداد سطح روشنایی است که در این قسمت برابر 256×256 است.

د)

```
brain.jpg features
Contrast = 114.3506
uniformity = 0.037457
Homogenity = 0.43554
Entropy = 10.3673
```

```
sample.png features
Contrast = 2.984
uniformity = 0.046276
Homogenity = 0.7526
Entropy = 7.1545
```

هیستوگرام دو تصویر متفاوت ممکن است عینا مثل هم شود و بنابراین ویژگی‌هایی که از آن استخراج می‌شود یکسان خواهد شد. اما با استفاده از GLCM احتمال کمی وجود دارد که برای دو تصویر متفاوت GLCMها یکسان شود. بنابراین ویژگی‌هایی که با

استفاده از GLCM به دست می‌ایند برای دو تصویر متفاوت با احتمال زیادی متفاوت خواهند بود و همانطور که دیده می‌شود ویژگی‌های به دست آمده برای دو تصویر داده شده بسیار متفاوت است، در حالیکه با استفاده از pdf ویژگی‌ها یکسان به دست آمد.

هـ) برای کاهش سایز ماتریس GLCM سطوح روشنایی را کوانتیزه می‌کنیم و تعداد سطوح را از ۲۵۶ به ۶۴ می‌رسانیم. در این صورت سایز ماتریس GLCM برابر 64×64 می‌شود. محاسبه ماتریس GLCM سنگین است به خصوص زمانیکه با تصاویر پزشکی (در مرتبه ۱ میلیون پیکسل) سر و کار داریم محاسبه این ماتریس سنگین و طاقت فرسا می‌شود. هم‌چنین سایز ۲۵۶ نیز بزرگ است و استخراج ویژگی به کندی صورت می‌پذیرد. بنابراین برای محاسبه سریع‌تر از ماتریس GLCM با سایز کوچک‌تر استفاده می‌شود.

و) قدرمطلق تفاضل دو سطح روشنایی بین دو پیکسل بصورت GLDM تعریف می‌شود[۲]. با در نظر گرفتن سطح روشنایی به صورت تابعی از مختصات تصویر به شکل $f(x,y)$، آنگاه برای جابه‌جایی به اندازه $\delta = (\Delta x, \Delta y)$ که در آن Δx و Δy اعداد integer هستند تفاضل grey levelها بصورت زیر تعریف می‌شود:

$$f_\delta(x,y) = |f(x,y) - f(x+\Delta x, y+\Delta y)|$$

در این روش δ بصورت $(0,d),(-d,d),(d,0),(-d,-d)$ تعریف می‌شود که δ فاصله inter-sample spacing است. در نهایت برای این ۴ فاصله مختلف، pdfها محاسبه می‌شوند و از روی این pdfها ۵ ویژگی حساب می‌شوند که عبارتند از:

a) Contrast

$$CON = \sum_{i=0}^{G-1} i^2 f'(i|\delta) \qquad (3.13)$$

b) Entropy

$$ENT = \sum_{i=0}^{G-1} f'(i|\delta) log f'(i|\delta) \qquad (3.14)$$

c) Mean

$$MEAN = \sum_{i=0}^{G-1} i f'(i|\delta) \qquad (3.15)$$

d) Angular second moment

$$ASM = \sum_{i=0}^{G-1} [f'(i|\delta)]^2 \qquad (3.16)$$

e) Inverse difference moment

$$IDM = \sum_{i=0}^{G-1} \frac{f'(i|\delta)}{i^2+1} \qquad (3.17)$$

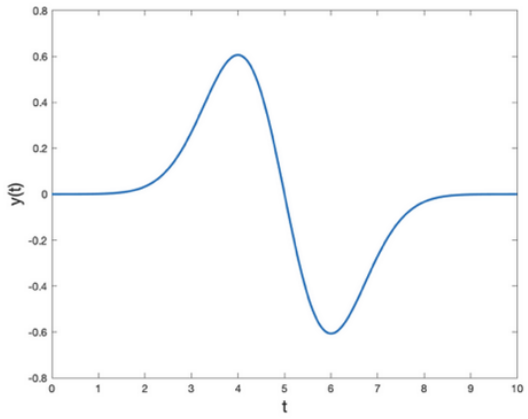
[2] Texture Feature-based Document Image Retrieval, Fahime Alaei

GLRM ماتریسی است که از آن برای استخراج ویژگی‌های texture استفاده می‌شود. یک grey level run بصورت یک خط از پیکسل‌ها با شدت روشنایی مشخص در یک جهت خاص تعریف می‌شود. تعداد این grey level runها همان grey level run length است و تعداد رخداد آن‌ها را run length value می‌نامیم. [۳]

[3] Texture Feature Analysis for Different Resolution Level of Kidney Ultrasound Images, Wan Nur Hafsha Wan Kairuddin and Wan Mahani Hafizah Wan Mahmud 2017, IOP Conf. Ser.: Mater. Sci. Eng. 226 012136

۳. الف) یکی از مشکلات DFT این است که اطلاعات فرکانسی global را استخراج می‌کند. به عبارت دیگر فرکانس‌هایی که بصورت پایدار در سیگنال حضور دارند. بنابراین در بسیاری از کاربردها (مانند پردازش سیگنال ECG) استفاده از DFT پاسخگو مساله نیست و از تبدیل ویولت استفاده می‌شود که در آن سیگنال به جای تصویر شدن روی انواع سیگنال‌های سینوسی روی موجک‌ها تصویر می‌شود.

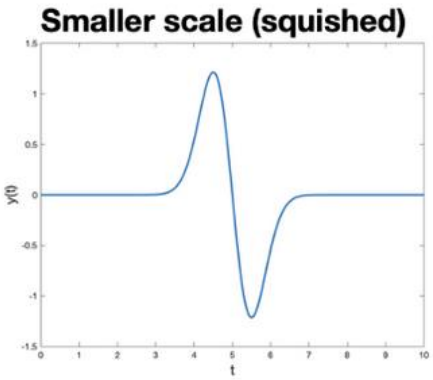
یک موجک، نوسانی موج گونه است که در مکان یا زمان جابه‌جا می‌شود. هر موجک دو ویژگی دارد که عبارتند از scale که فشرده یا باز بودن موجک را تعیین می‌کند و location که محل موجک را در زمان یا مکان تعیین می‌کند.

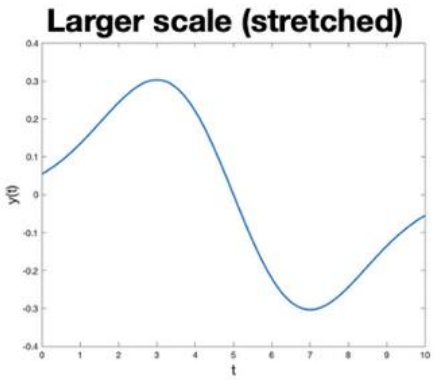


$$-(x-b)e^{\frac{-(x-b)^2/(2a^2)}{\sqrt{2\pi}a^3}}$$

First derivative of Gaussian Function

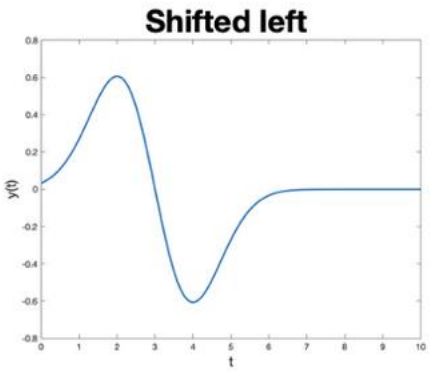
در شکل بالا پارامتر a باز و بسته بودن ویولت را تعیین می‌کند. هر چه مقدار آن را کوچکتر انتخاب کنیم، ویولت جمع تر می‌شود و برعکس.

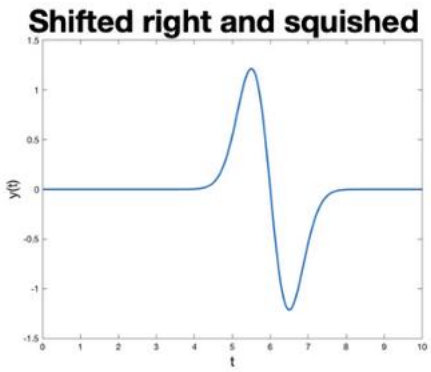




a

پارامتر b محل موجک را تعیین می‌کند و با کاهش آن به سمت چپ می‌رود و برعکس.

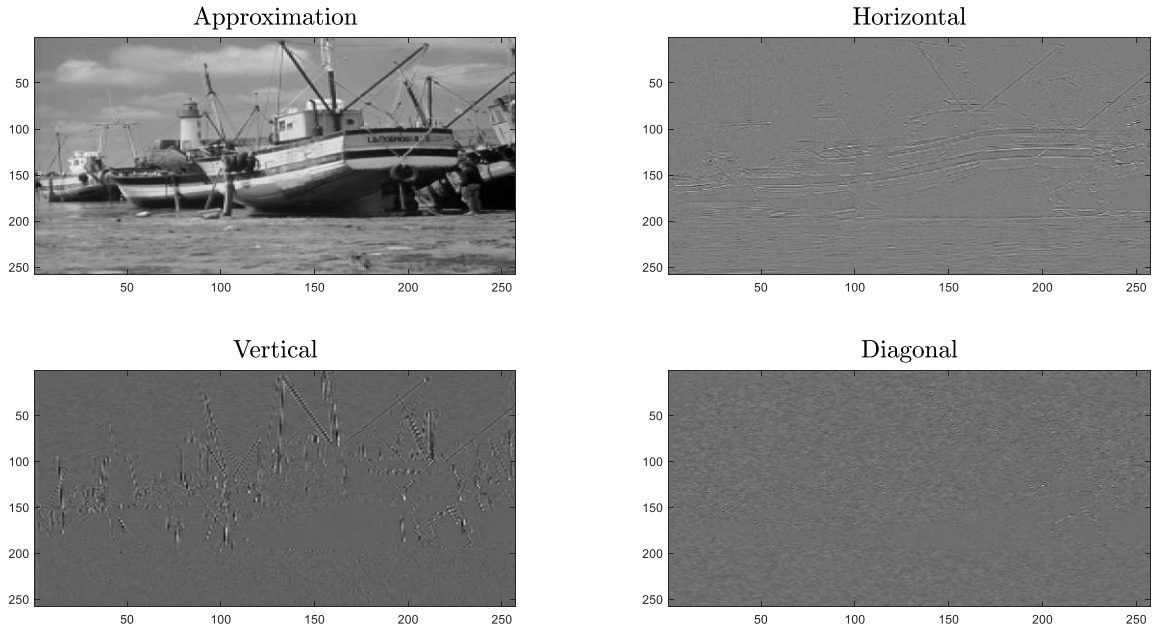




b

در واقع می‌خواهیم بدانیم که چقدر از هر موجک در سیگنال موجود است که از این نظر بسیار شبیه به مفاهیم DFT است. بنابراین سیگنال موجک با سیگنال اصلی کانوالو می‌شود و به این ترتیب ضرایب به دست می‌آیند. [۴]

ب)



ج)

```
reconstruction error = 20.8285
```

[4] https://towardsdatascience.com/the-wavelet-transform-e9cfa85d7b34

reconstructed image

original image

د)

Wavelet Transform for Image Compression

original image

reconstructed image (95%)

reconstructed image (40%)

reconstructed image (5%)

```
reconstruction error (95%) = 20.6407
reconstruction error (40%) = 20.4699
reconstruction error (5%) = 113.3241
```

هـ)

```
using wavelet transform
Compression Ratio (95%) = 1.0204:1
Compression Ratio (40%) = 1.0988:1
Compression Ratio (5%) = 3.0561:1
```

Fourier Transform for Image Compression

original image

reconstructed image (95%)

reconstructed image (40%)

reconstructed image (5%)

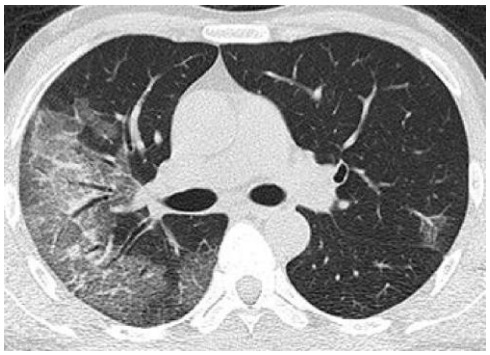
```
Compression Ratio (95%) = 1.0657:1
Compression Ratio (40%) = 1.1059:1
Compression Ratio (5%) = 1.3527:1
```

تعداد ضرایب DFT و DWT در مرتبه یکسانی (حدودا 26k) به دست آمد. مشاهده می‌شود اگر درصد یکسانی از ضرایب را نگه داریم، عملکرد DFT حدودا بهتر است. به ازای ۵٪ در DWT عملا چیزی از تصویر باقی نمانده و بنابراین این فشرده سازی چندان کارآمد نخواهد بود. در کل به نظر می‌رسد از نظر ذخیره‌سازی DFT عملکرد بهتری دارد.

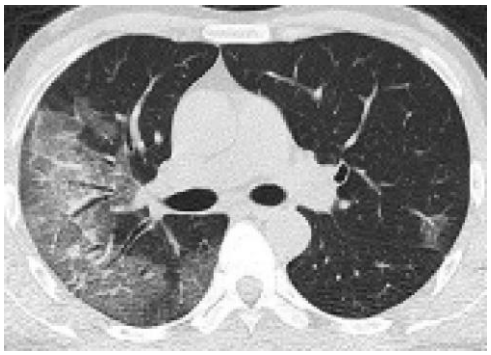
۴. الف) برای این کار از تابع visuShrink.m استفاده کردیم که در آن آستانه برای هر subband بصورت جداگانه تعیین می‌شود. در ادامه با استفاده از hard thresholding عمل حذف نویز را انجام می‌دهیم.

ب)

noisy image



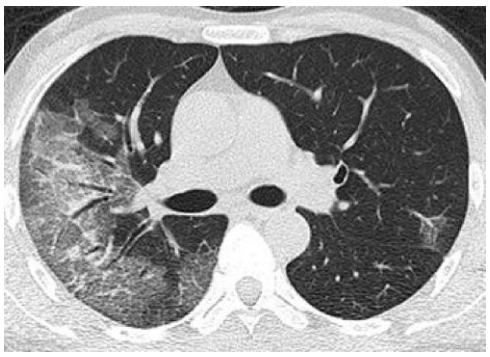
denoised image (using DWT)



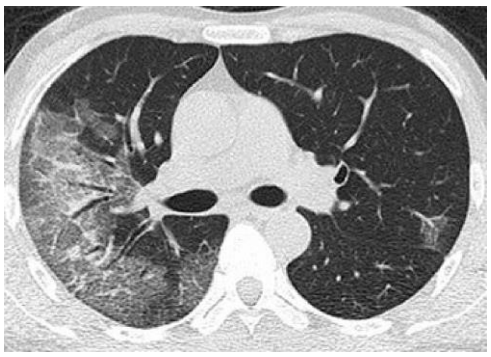
عمل حذف نویز در برخی نواحی از تصویر به خصوص در ناحیه استخوانی (مایل به سفید) مشهود است.

ج)

noisy image



denoised image (using DFT)



مشاهده می‌شود DFT برای حذف نویز این تصویر چندان مناسب نیست. از DFT می‌توان برای تصاویری استفاده کرد که psd محتوای تصویر سطح بالاتری نسبت به نویز جمع شونده (معمولا سفید) دارد و در چنین مواردی می‌توان با یک آستانه گذاری ساده و جداکردن محتوای فرکانسی مطلوب، تصویر اصلی را بازسازی کرد. اما در مواردی که نویز پیچیده تر است می‌توان با انتخاب موجک مناسب ابتکار عمل بیشتری داشت.