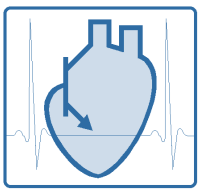


دانشگاه صنعتی امیرکبیر

دانشکده مهندسی پزشکی

گزارش تمرین چهارم پردازش تصویر

دانشجو

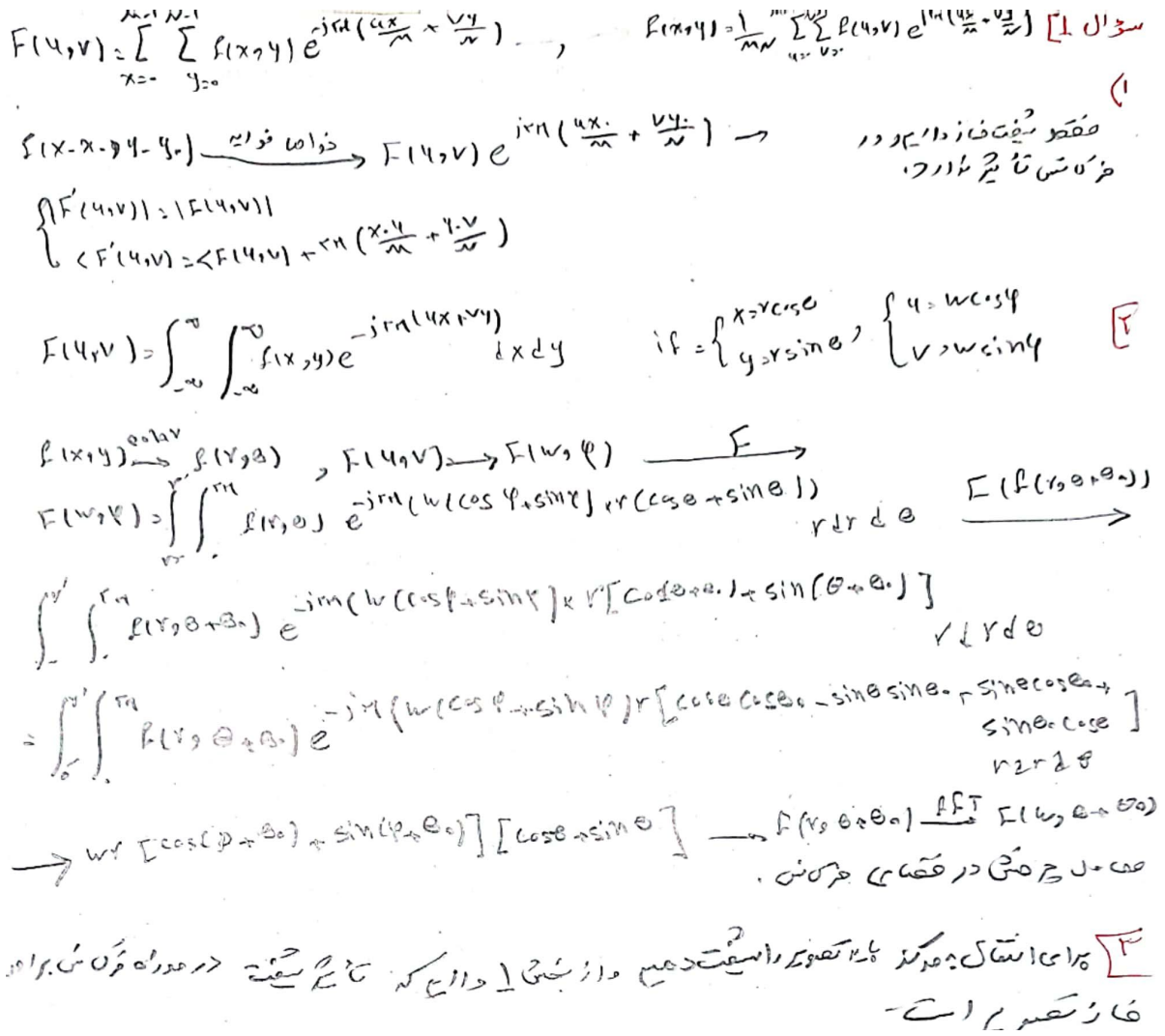
سیدابوالفضل مرتضوی ۹۸۳۳۰۶۳

آذرماه ۱۴۰۱

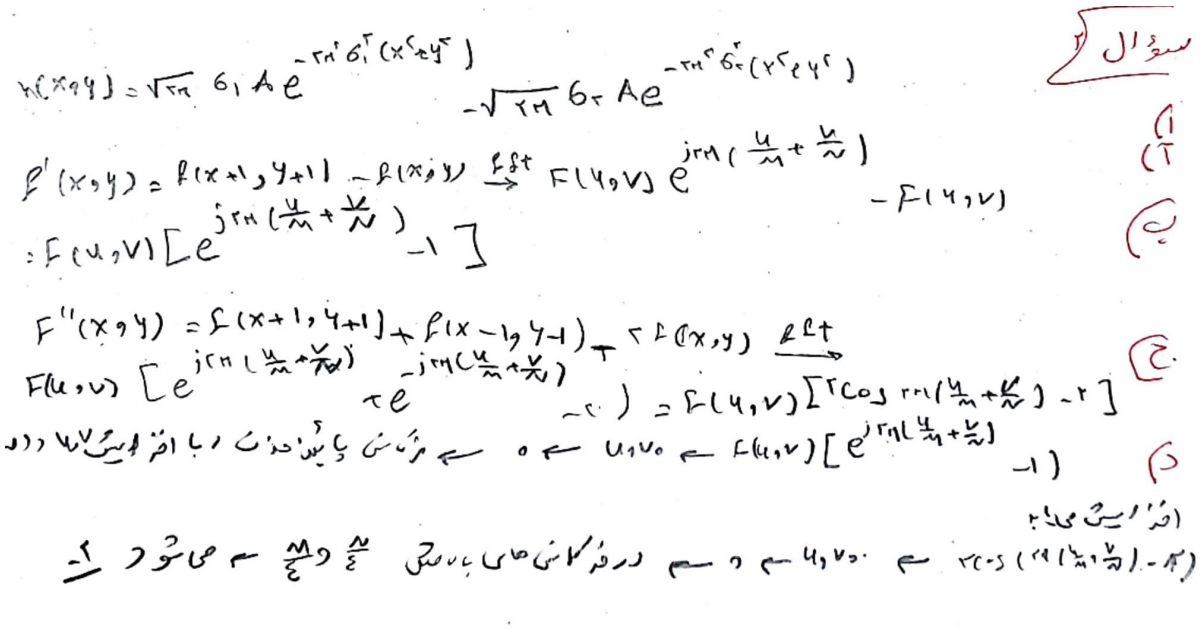
فهرست مطالب

فهرست مطالب

تمرین اول)



تمرین دوم)



تمرین سوم)

۳-۱ بخش۱)

```
created_img=np.zeros([200,200]).astype('uint8')
created_img[80:120,60:140]=255
```

با کد بالا این کار انجام شد.

۳-۲ بخش۲)

```
tranlate_matrix_x=np.float32(([1,0,20],[0,1,0]))
translated_img_x=cv.warpAffine(created_img,tranlate_matrix_x,(200,200))
tranlate_matrix_y=np.float32(([1,0,0],[0,1,-40]))
translated_img_y=cv.warpAffine(created_img,tranlate_matrix_y,(200,200))
rotate_matrix_30=cv.getRotationMatrix2D((100,100), 30, 1.0)
rotated_img_30=cv.warpAffine(created_img, rotate_matrix_30, (200,200))
rotate_matrix_90=cv.getRotationMatrix2D((100,100), 90, 1.0)
```

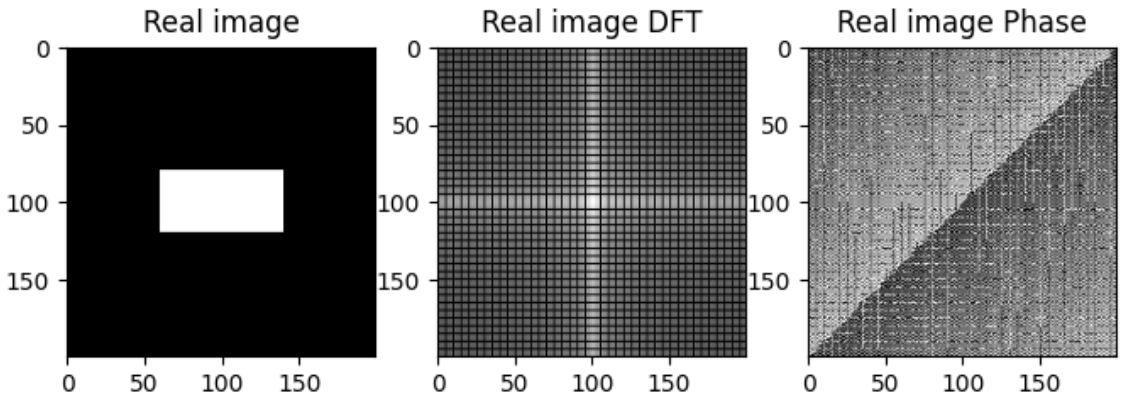
```
rotated_img_90=cv.warpAffine(created_img, rotate_matrix_90, (200,200))
```

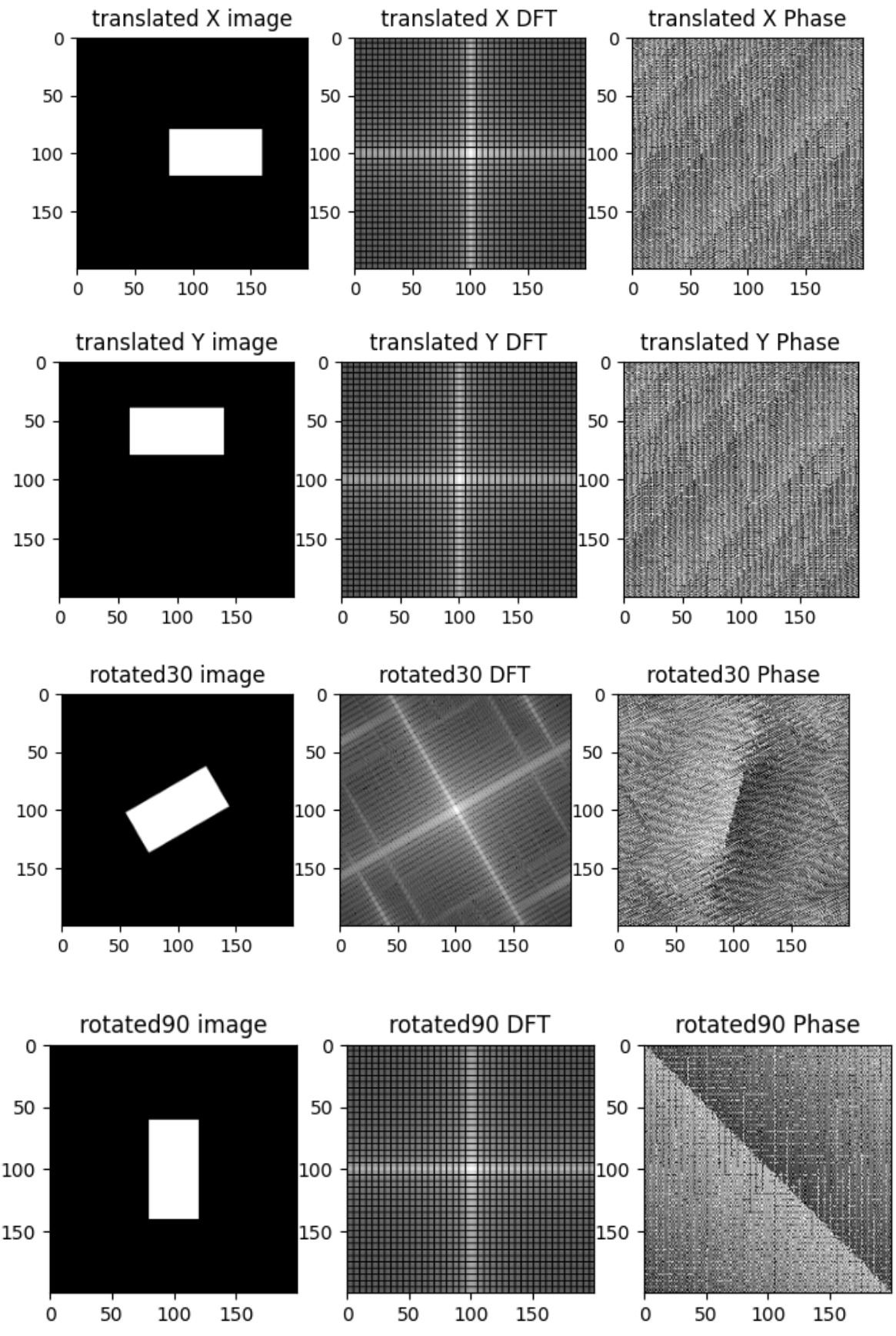
با استفاده از کد بالا تصاویر خواسته‌شده ایجاد شدند.

۳-۳ بخش۳)

```
real_img_dft=np.fft.fft2(created_img)
real_img_dft=np.fft.fftshift(real_img_dft)
real_img_phase=np.angle(real_img_dft)


trx_img_dft=np.fft.fft2(translated_img_x)
trx_img_dft=np.fft.fftshift(trx_img_dft)
trx_img_phase=np.angle(trx_img_dft)


try_img_dft=np.fft.fft2(translated_img_y)
try_img_dft=np.fft.fftshift(try_img_dft)
try_img_phase=np.angle(try_img_dft)

rot30_img_dft=np.fft.fft2(rotated_img_30)
rot30_img_dft=np.fft.fftshift(rot30_img_dft)
rot30_img_phase=np.angle(rot30_img_dft)


rot90_img_dft=np.fft.fft2(rotated_img_90)
rot90_img_dft=np.fft.fftshift(rot90_img_dft)
rot90_img_phase=np.angle(rot90_img_dft)
```

بااستفاده از کد بالا تبدیلات خواسته‌شده انجام شدند و خروجی درشکل ۳.۱ نشان داده شده است.





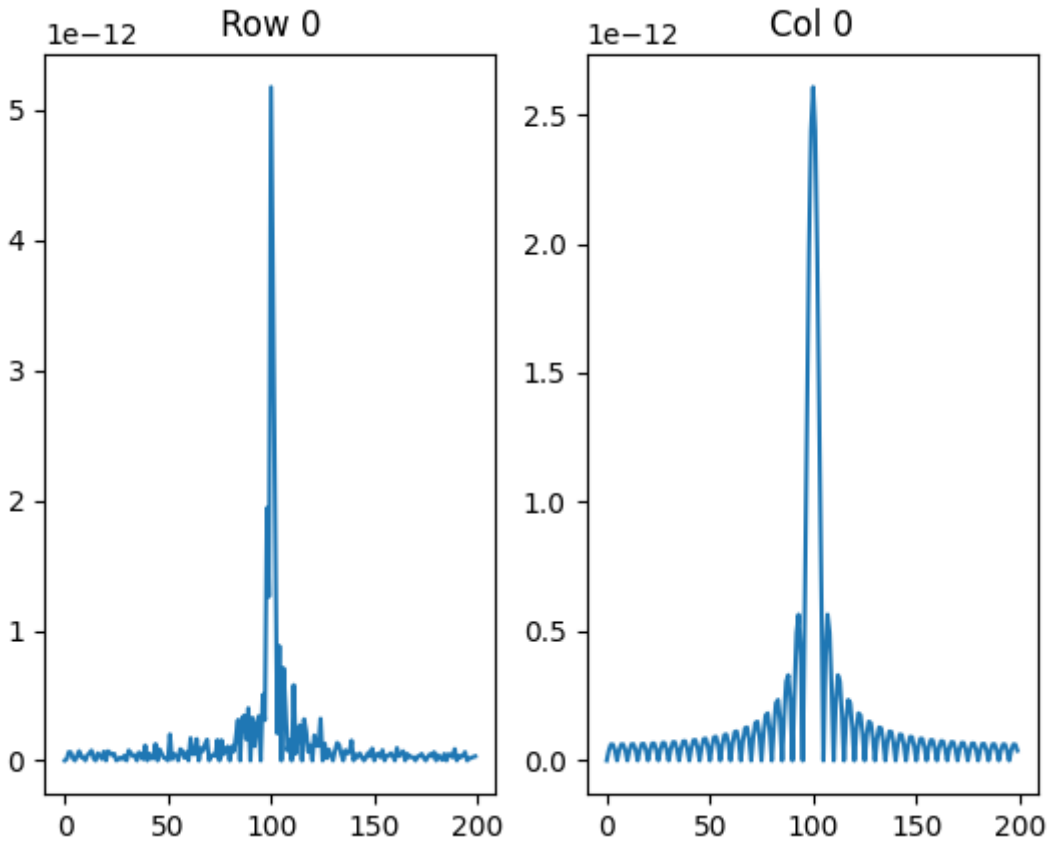
شکل ۳.۱ تصاویر خروجی بخش سوم

همانطور که در تصاویر بالا مشخص است حرکت درراستای X,Y تغییری در فضای فرکانسی ایجاد نمی‌کند اما در فاز باعث تغییرات می‌شود که این تغییرات به خاطر ضرب شدن e^{jx} است. چرخش تصویر باعث تغییر در فضای فرکانسی و فضای فاز می‌شود. در چرخش ۹۰ درجه به علت تقارن شکل فرکانسی تغییری در شکل دیده‌نمی‌شود.

۳-۴ بخش۴)

```
real_img_row0=np.abs(real_img_dft[0])
real_img_col0=np.abs(real_img_dft[:,0])
```

بااستفاده از کد بالا این کار انجام شد و خروجی درشکل ۳.۲ نشان داده شده است.

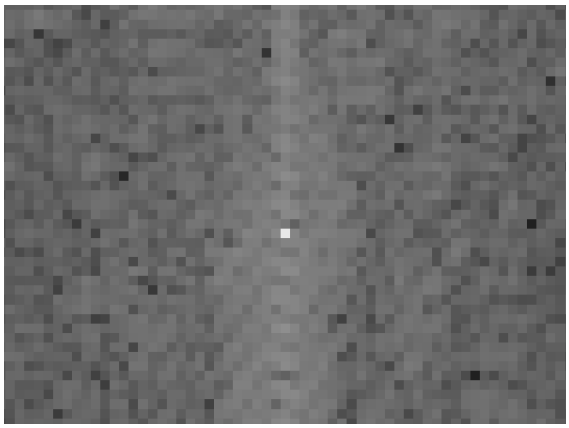


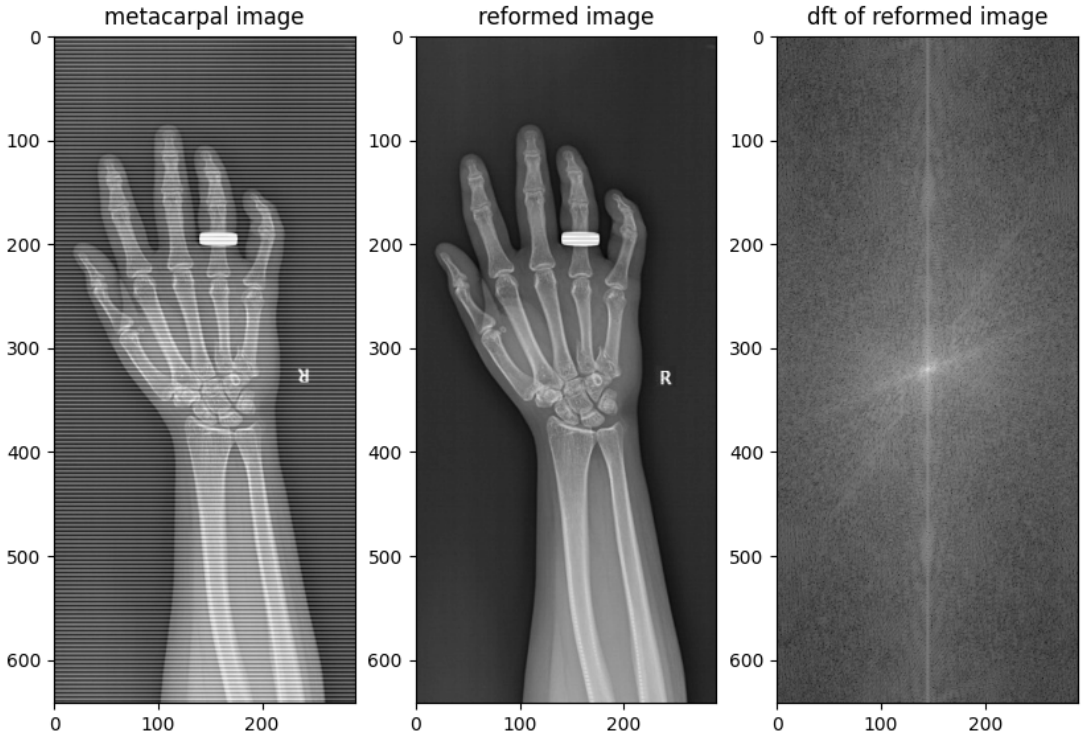
شکل ۳.۲ خروجی بخش د

۳-۵ بخش‌های ۵تا۱۰

```
meta_img=cv.imread('metacarpal.png',0)
meta_fft=np.fft.fft2(meta_img)
meta_fft=np.fft.fftshift(meta_fft)

meta_fft_max=abs(meta_fft[0:200,0:200])
meta_fft_max=list(meta_fft_max)
max_val=0
max_cor=(0,0)
for i in range(200):

    if max(meta_fft_max[i])>max_val:
        max_val=max(meta_fft_max[i])
        max_cor=(i,list(meta_fft_max[i]).index(max(meta_fft_max[i])))
cor_x,cor_y=max_cor
meta_fft[cor_x,cor_y]=0
meta_fft[642-cor_x,cor_y]=0


reformed_img=np.fft.ifftshift(meta_fft)
reformed_img=abs(np.fft.ifft2(reformed_img))


R_non_rot=reformed_img[318:338,230:250]
R_non_rot_dft=np.fft.fft2(R_non_rot)
R_non_rot_dft=np.fft.fftshift(R_non_rot_dft)

R_rot_dft=R_non_rot_dft[20::-1,20::-1]

R_rot=np.fft.ifftshift(R_rot_dft)
R_rot=abs(np.fft.ifft2(R_rot))

reformed_img[318:338,230:250]=R_rot


reformed_img_dft=np.fft.fft2(reformed_img)
reformed_img_dft=np.log1p(np.abs(np.fft.fftshift(reformed_img_dft)))
```

بااستفاده از کد بالا کارهای خواسته شده انجام شد. برای حذف نویز دو نقطه سفید در مختصات (160,145) و (482,145) وجود داشتند که صفر شدند و در تصویر خطوط حذف شدند. خروجی نهایی درشکل ۳.۳ نشان داده شده است.





شکل ۳.۳ خروجی نهایی

سوال چهارم)

۴-۱ بخش ۱)

```
shoulder_img=cv.imread('shoulder.jpg',0)
```

۴-۲ بخش۲)

```
shoulder_img=cv.imread('shoulder.jpg',0)

zero_pad_matrix=np.zeros([2048,2048])

zero_pad_matrix[0:1024,0:1024]=shoulder_img

zero_pad_shoulder=zero_pad_matrix

centered_shoulder=center_img(zero_pad_shoulder)

shoulder_img_dft=np.fft.fft2(centered_shoulder)
```

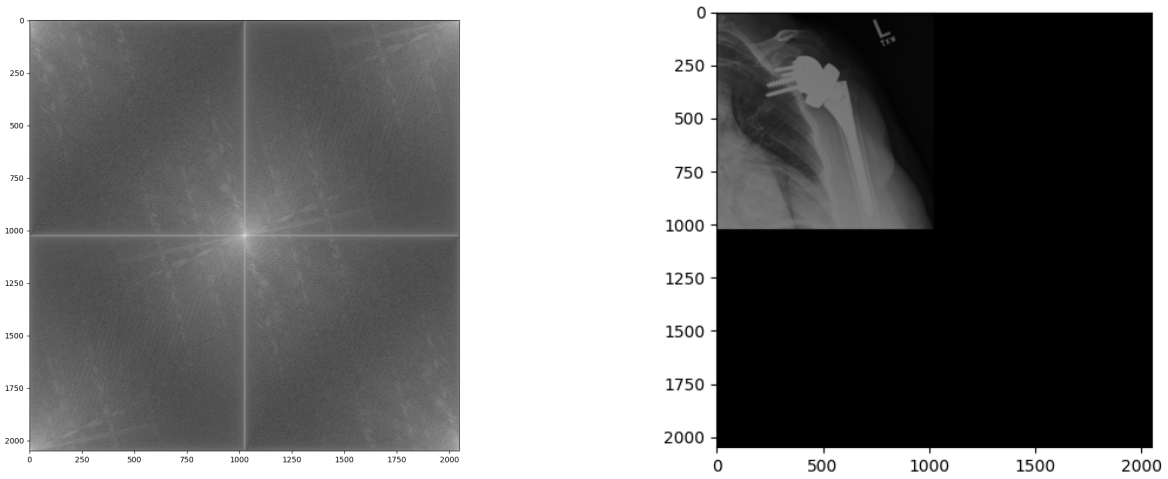
در این بخش ابتدا تصویر به اندازه ۱۰۲۴پیکسل از هرطرف پد شد و سپس با تابع center_img تصویر برای انتقال به مرکز در حوزه فرکانس آماده شد.

```
def center_img(img):
    centered_img=np.zeros([img.shape[0],img.shape[1]])
```

```
    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            centered_img[i,j]=img[i,j]*((-1)**(i+j))
            if centered_img[i,j]<0:
             if centered_img[i,j]<0:

                centered_img[i,j]=0
   return centered_img
```

دراین تابع ابتدا تمامی مقادیر در $(-1)^{x+y}$ ضرب شدند و سپس مقادیر کوچکتر از صفر به صفر مپ شدند.

خروجی این مراحل درشکل ۴.۱ نشان داده شده است.



شکل ۴.۱ تصاویر خروجی این بخش

۴-۳ بخش ۳)

```
def ideal_fil(M,N,cut_fer):
    H=np.zeros([M,N]).astype('float')
    for u in range(M):
        for v in range(N):
            D=np.sqrt((u-M/2)**2+(v-N/2)**2)
            if D>cut_fer:
                H[u,v]=0
            else :
                H[u,v]=1
    return H
def butter_fil(M,N,n,cut_fer):
    H=np.zeros([M,N]).astype('float')
    for u in range(M):
        for v in range(N):
            D=np.sqrt((u-M/2)**2+(v-N/2)**2)
            H[u,v]=1/(1+pow((D/cut_fer),2*n))
```

```
    return H
def Gaussian_fil(M,N,cut_fer):
    H=np.zeros([M,N]).astype('float')
    for u in range(M):
        for v in range(N):
            D=np.sqrt((u-M/2)**2+(v-N/2)**2)
            H[u,v]=exp((-1*pow(D,2))/(2*pow(cut_fer,2)))
    return H
```

در این بخش توابع خواسته شده پیاده سازی شدند. در تمامی توابع M,N ابعاد تصویر و cut_fer شعاع است. در فیلتر باترورث n درجه فیلتر است.

۴-۴ بخش۴)

```
row,col=shoulder_img.shape
ideal_HP_30=1-ideal_fil(2*row,2*col,30)
ideal_HP_100=1-ideal_fil(2*row,2*col,100)

butter_HP_30=1-butter_fil(2*row,2*col,2,30)
butter_HP_100=1-butter_fil(2*row,2*col,2,100)

gaussian_HP_30=1-Gaussian_fil(2*row,2*col,30)
gaussian_HP_100=1-Gaussian_fil(2*row,2*col,100)
```

در این قسمت توابع بالاگذر پیاده سازی شدند. علت استفاده از ضریب ۲ در ابعاد، به خاطر پد کردن تصاویر است که ابعاد را بالا برده است.

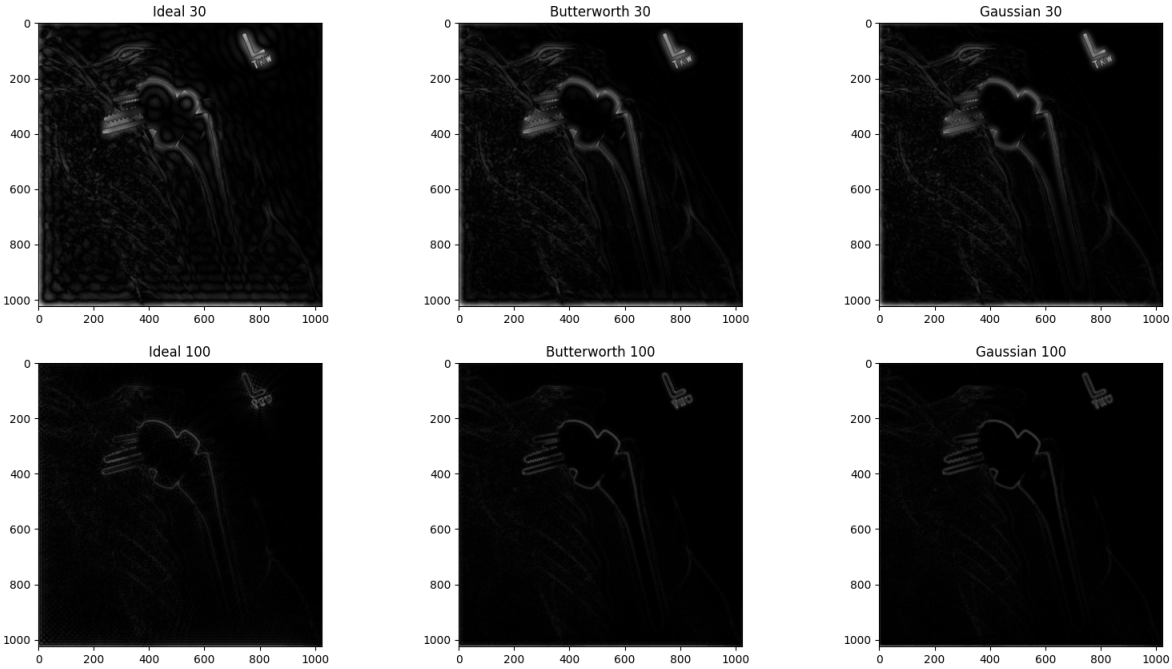
```
image_ideal_30=ideal_HP_30*shoulder_img_dft
image_ideal_100=ideal_HP_100*shoulder_img_dft

image_butter_30=butter_HP_30*shoulder_img_dft
image_butter_100=butter_HP_100*shoulder_img_dft

image_gauss_30=gaussian_HP_30*shoulder_img_dft
image_gauss_100=gaussian_HP_100*shoulder_img_dft
```

دراینجا فیلترها برروی تصاویر اعمال شدند.

```
'''ideal'''

shoulder_IHP_30=np.fft.ifft2(image_ideal_30)
shoulder_IHP_30_show=np.abs(shoulder_IHP_30)
shoulder_IHP_30_show=shoulder_IHP_30_show[0:1024,0:1024]

shoulder_IHP_100=np.fft.ifft2(image_ideal_100)
```

```
shoulder_IHP_100_show=np.abs(shoulder_IHP_100)
shoulder_IHP_100_show=shoulder_IHP_100_show[0:1024,0:1024]

'''butterworth'''

shoulder_BHP_30=np.fft.ifft2(image_butter_30)
shoulder_BHP_30_show=np.abs(shoulder_BHP_30)
shoulder_BHP_30_show=shoulder_BHP_30_show[0:1024,0:1024]

shoulder_BHP_100=np.fft.ifft2(image_butter_100)
shoulder_BHP_100_show=np.abs(shoulder_BHP_100)
shoulder_BHP_100_show=shoulder_BHP_100_show[0:1024,0:1024]




'''gaussian'''

shoulder_GHP_30=np.fft.ifft2(image_gauss_30)
shoulder_GHP_30_show=shoulder_GHP_30_show[0:1024,0:1024]
shoulder_GHP_100=np.fft.ifft2(image_gauss_100)
shoulder_GHP_100_show=np.abs(shoulder_GHP_100)
shoulder_GHP_100_show=shoulder_GHP_100_show[0:1024,0:1024]
```

در این بخش بازگردانی تصاویر به حوزه مکان و برش و حذف بخش پد شده انجام شده است.خروجی تمام مراحل درشکل ۴.۲ نشان داده شده است.



شکل ۴.۲ خروجی این بخش